PRACTICAL AND EXPERIMENTAL APPROACHES TO INFORMATION SECURITY EDUCATION

PRACTICAL AND EXPERIMENTAL APPROACHES TO INFORMATION SECURITY EDUCATION

Workshop Chair

Cynthia Irvine

Proceedings Production by

Matthew Rose and Naomi Falby
Naval Postgraduate School, Monterey, California, USA

Center for Information Systems Security Studies and Research
Naval Postgraduate School
Monterey, California

Seventh Workshop on Education in Computer Security (WECS7)

Naval Postgraduate School, Monterey, California
04-06, January 2006

WECS is dedicated to the advancement of Information Assurance education. CISR launched WECS in 1999 and is proud to have started such a successful tradition of outreach to the computer security community.

The goal of WECS7 is to bring together members of the information assurance education community to promote an exchange of ideas related to shared educational objectives.

Library of Congress Cataloging-in-Publication Data
A C.I.P. Catalogue record for this book is available from the Library of Congress.

Practical and Experimental Approaches to Information Security Education

Edited by Cynthia Irvine, Matthew Rose and Naomi Falby
ISBN 0-9755139-1-5

Printed in the United States of America.

Please visit: http://cisr.nps.navy.mil

Contents

Contributing Authors

Shiva Azadegan, Matt Bishop, Marina A. Cappellino, Chris Eagle, Simone Fischer-Hübner, Judith L. Gersting, Hans Hedbom, Houssain Kettani, Stefan Lindskog, Leonardo A. Martucci, Casey W. O'Brien, Jide B. Odubiyi, Michael O'Leary, Stephen Providence, Andrew L. Reifers, Scott J. Roberts, Rose Shumba, Carol Taylor, Mike Thompson, Percy Vega, James Walden, Daniel F. Warren, Alexander Wijesinha, Jinsheng Xu, Huiming Yu, Xiaohong Yuan, and Marius Zimand

Preface

Dear Friends and Colleagues in Information Security Education

Since the last Workshop on Education in Computer Security, we have witnessed a greater level of appreciation within the general population regarding the protection of information. Perhaps the clearest motivating factor for this is the threat of identity theft. Unfortunately, even though many guidelines regarding "safe" computing have been published in the popular media, the possibility of falling victim to cyber crime remains high. Part of this is due to the increased sophistication of the attackers, but a major factor is the lack of user education.

As educators we face numerous challenges: the education of professionals who will develop and maintain future security solutions, the education of other information technology professionals, and the education of the general population. This workshop will touch on each of these groups. To achieve this, WECS serves two objectives. First, it is a forum to discuss emerging concepts in information assurance education, and, second, it is intended to provide participants with practical experience in developing educational materials for use with CyberCIEGE, an extensible cyber security video game.

Educators from across the nation have written excellent papers that are both informative and thought provoking. We learn how information assurance can be incorporated into an undergraduate educational program, how attack tree modeling can provide a teaching framework, the challenges associated with teaching vulnerability analysis, and the use of visualization tools in laboratory exercises. Several broad topics relating to the teaching of cyber ethics and context, as well as future directions in information assurance education, provide grist for lively discussion.

The keynote speakers for the workshop address two interesting topics. The first is system forensics. Its treatment includes both traditional and non-traditional applications of forensic techniques. Our second keynote talk describes how checklists can help educators teaching information assurance. These can be used to help students analyze problems, to review the security of operational systems, and as a means to ensure that educational objectives are met.

A feature of this year's workshop is CyberCIEGE. We are especially grateful to Michael Thompson, one of the principle developers of CyberCIEGE, for providing several afternoons of detailed laboratory exercises that will allow participants to create and tailor game scenarios for use in their classes. As part of this activity, Rivermind Inc. has made special arrangements for non-government WECS participants to have access to the underlying CyberCIEGE simulation.

We hope that the readers of these proceedings will join the conference organizers, authors and participants for future workshops and conferences on information security education.

We are grateful for the work of the members of the WECS committees who volunteered their time. The complexities of conference mechanics can be daunting. Thanks to the behind the scenes preparations of Naomi Falby, we can look forward to a beautifully orchestrated workshop. We are also grateful to Matthew Rose whose attention to detail has again led to the creation of a carefully constructed proceedings. We are also grateful for the laboratory support provided by Phil Hopfner.

January 2006
Cynthia Irvine

COMPUTER FORENSICS IN THE CLASSROOM

Keynote

Chris Eagle

Naval Postgraduate School

Digital forensics is one of the fastest growing areas within the computer security field today. Digital devices continue to penetrate every aspect of our lives and digital crimes continue to become more and more sophisticated.

The resulting demand for trained computer forensics analysts presents unique challenges to a computer science degree program. The computer forensics process is traditionally divided into five phases: preparation, incident identification/response, evidence collection, evidence analysis, and presentation of findings. The hard "computer science" within this process lies primarily in the aforementioned evidence collection and analysis phases, yet it would be a disservice to our students to avoid the three remaining phases. In this talk I will discuss the challenges of presenting a one quarter course in computer forensics that is both sufficiently broad to cover all of the requisite phases and sufficiently deep to provide the student with a solid scientific foundation in the field.

Chris Eagle is the Associate Chairman of the Computer Science Department at the Naval Postgraduate School (NPS) in Monterey, CA. A computer engineer/scientist for 20 years, his research interests include computer network operations, computer forensics and reverse/anti-reverse engineering. He has been a trainer and speaker at conferences such as Blackhat, Codecon and Shmoocon and is a co-author of "Gray Hat Hacking, The Ethical Hacker's Handbook". In his spare time, he is the benevolent dictator of the Skewl Of Rewt, past champions of the annual Capture the Flag competition at Defcon.

UNDERGRADUATE COMPUTER SECURITY EDUCATION

A Report on our Experience and Learning

Shiva Azadegan, Michael O'Leary, Alexander Wijesinha, and Marius Zimand

Towson University

Abstract: We describe our experiences from the first three years we have offered our track in computer security for our computer science major. We present the details of the track, including descriptions of the courses we have offered. We discuss the lessons we have learned offering the track, as well as the challenges that remain

Key words: Computer Security Education

1. INTRODUCTION

In Fall 2002 we offered our first course in our new Computer Security Track in the Computer Science major here at Towson University, and last year we graduated our second class of students. In this paper, we would like to describe our track and how it has developed as the program has grown.

Our track in computer security is our traditional Computer Science major where our students choose specific courses in computer security for their upper level electives. The computer security portion of the track is centered on the following seven courses:

- Computer Ethics,
- Introduction to Information Security,
- Introduction to Cryptography,
- Network Security,
- Application Software Security,
- Operating Systems Security, and
- Case Studies in Computer Security.

By design, our track focuses on the technical, practical, and applied areas of computer security. This design decision was made in part because this is a track within our regular computer science major; our students still take the core computer science courses like Computer Architecture, Operating Systems, and Database Management. The track courses are actually built upon the core courses. Figure 1 depicts the prerequisite tree for these courses. We also have a Master's degree with a concentration in computer security, but we will not discuss that concentration in this paper.

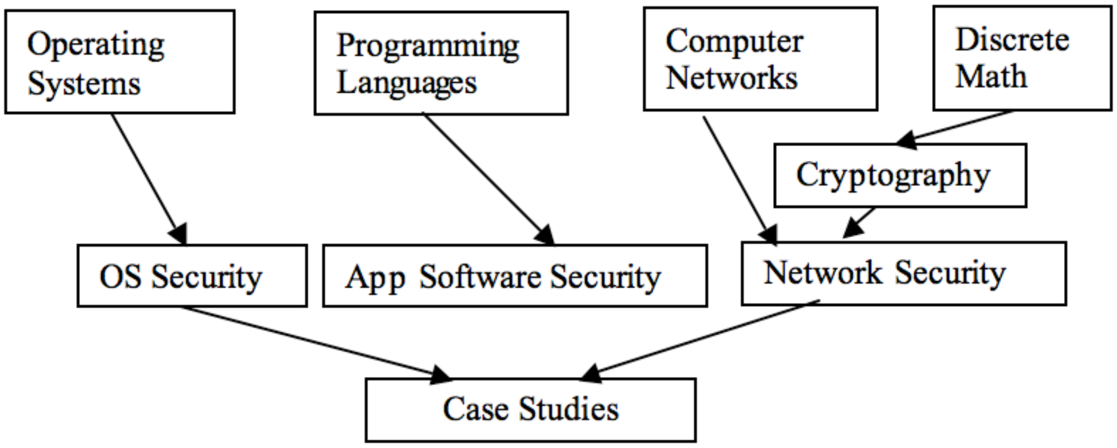


Figure 1. Computer Security Track Prerequisite Tree

Our initial approach to the design of our track can be found in [3,4]. Since then, a number of other approaches to integrating security into the curriculum have been presented. Perrone, Aburdene and Meng [15] present an informal survey of tracks and courses in computer security, while Tikekar [22] describes a new undergraduate track in computer security and information assurance at Southern Oregon University; see also [21]. Similarly, Dornseif *et. al*. [8,9] describe a data security program at a German university. A case study of the general process of information security curriculum development is presented by Bogolea and Wijekumar in [7].

As we enter the fourth year of existence for our computer security track here at Towson University, we have found that enrollment in our track has been healthy though small for a school of this size and has increased each year. Enrollment in our Cryptography course has increased from 15 in the track's first year, to 17 in the second to 21 last year. Enrollment in this course is an important barometer of the number of students who are entering the security track, as it is a prerequisite for the Network Security course, which in turn is a prerequisite for the Case Studies course; thus we recommend students in the track take this course early in their junior year. At the other end of the track is the Case Studies course, which is our capstone experience for the track, and taken in the spring semester of the senior year. Enrollment in this course was 5 in 2004, and nearly doubled to 9 in 2005; we expect that enrollment in 2006 will be even larger. Graduates from our track have been in great demand, and our former students have had little difficulty finding industry jobs quite quickly. Employers have been so pleased with our graduates, that they have contacted the faculty here directly, asking us to help them recruit more of our graduates.

2. FEATURES CONTRIBUTED TO THE SUCCESS OF THE TRACK

Based on our course evaluations and feedback from some of the employers of our students, the strong hands-on component of the security track has been the most valuable learning experience for our students. The courses in the track are built upon the core courses in the computer science program and they all have upper-level computer science courses as their prerequisite. Thus, students do have the theoretical knowledge, the maturity and good programming skills that are necessary to do elaborate and interesting applied projects. Though, developing such projects is very time consuming for the instructors, we believe that is imperative for any computer security training.

We also made the decision that the track to be focused and technical for the CS majors rather than broad and open to all; thus the students after graduating are skilled, confident and knowledgeable enough to be hired as information security officers.

All of the courses in the computer security track use our dedicated computer security laboratory for lab exercises. This is a physically secured room with a local network that is isolated, both from the campus as well as from the Internet. Student access to the laboratory is allowed outside of class only to students registered in one of our security courses. There are a number of different approaches one can take to the design of an isolated security laboratories; we mention [11, 14, 17, 18, 24, 25, 26]. See also [10, 27] for distributed security laboratories.

The lab itself contains 28 high end workstations running VMWare, and students do their lab work on virtual machines. One of the prime advantages of this approach is that it lets us use the classroom laboratory for a range of courses. Each class maintains their own set of virtual machines; changing the class is as simple as changing the virtual machines that are running. These virtual machines give our students the ability to experiment with a wide range of operating systems; We have used Windows 2000, Windows XP, Red Hat Linux, SUSE Linux, CentOS, and FreeBSD. VMWare also allows users to run more than one virtual machine on a host at the same time; we have run as many as six virtual machines simultaneously on one host. Further, because VMWare has the ability to set up virtual networks for the virtual machines on a host, we can let students set up their own small networks without the need for additional hardware.

Our choice of a flexible laboratory based on virtual machines has some disadvantages however. For example, we have not been able construct complex network topologies for the live exercises in our case studies course that you will find in, for example, [14, 18]. We have also limited ourselves to Intel computers; we do not have virtual machines that simulate a Mac or network hardware like a router or switch. Despite these limitations, the security laboratory has been an essential component of the early success of our track.

3. THE EXPERIENCES & LEARNING

In this section we share our experiences and learning and discuss the changes that were consequently made to the track. As explained in reference [3,4], two of the courses in the track, Computer Ethics and Introduction to Information Systems Security, were existing courses. The former is a required course for all our students and the latter is an upper-level elective course. This section focuses on the remaining five courses in the track that were developed and taught by the authors.

3.1 Cryptography

The Cryptography course is, roughly speaking, structured into three modules: (1) symmetric cryptography, (2) public-key cryptography, and (3) protocols and other applications. The course has been offered four times in the Fall semesters from 2002 to 2005 and the enrollment has grown from 12 to 21 students. The required textbook is *Introduction to Cryptography with Coding Theory* by Wade Trappe and Lawrence C. Washington [23]. In what follows we focus on several teaching aspects that we think require special attention.

Within the first module, some classical cryptosystems are covered such as: shift, affine, substitution, Vigenere, Hill, one-time pad ciphers (the module also discusses the most commonly used modern cryptosystems, DES, with its variants, and AES). We found it appropriate to use a relatively slow teaching pace in the exposition of classical cryptosystems (2.5 weeks). The simplicity of these systems allows the student to obtain a good understanding of the main security issues that a cryptosystem has to address, of the capabilities of different types of attacks, and of some basic principles that stand behind the design of any cryptosystem. This section of the course is also used to gently introduce some mathematical techniques, such as modular arithmetic and some notions from probability theory. Even though elementary, many students are not familiar with such mathematical tools, and they have the chance to see concrete applications of what otherwise may appear to be just mathematical abstractions.

We consider important that students understand and see the necessity of using rigorous mathematical formalism for the definition of security and secrecy as opposed to an ad-hoc and intuitive approach that does not provide any guarantees and is flatly hazardous in the context of cryptography. This is not easy to teach, however the one-time pad cipher allows for the exposition of *Shannon secrecy* and from here the instructor can make the transition to *statistical security*, and then to *computational security*. These notions, presented in the simple form of security against ciphertext-only attack, can be justified and illustrated with the classical cryptosystems that the students have learned.

We have attempted to alleviate the feeling that the course is math-heavy. For this reason, there are no long compact periods of time completely dedicated to mathematics. The mathematical notions are always introduced in the context of their utilization in cryptography systems and protocols. For example finite fields are introduced just after AES, the necessary elements of number theory are presented in conjunction with RSA, El-Gamal, Diffie-Hellman protocol, and so on.

The last module of the course on protocols and applications (such as bit commitment, zero-knowledge proofs, secret sharing, digital cash, electronic voting, coin flipping by telephone, etc.) should be very attractive for the students, since they apparently imply “impossible” things rendered feasible through clever combinations of rather elementary mathematical notions that have been presented earlier in the course. Unfortunately, this module is rushed at the end of the course when students focus more on their project and on the preparation for the final exam. Consequently we never had the time to cover but one or two applications from the above list and we felt that the students did not digest all the subtleties of these applications. We think that it is possible to teach parts of the second module more aggressively and to insert some of the applications at different points in the course.

The assignments, generally speaking, fall into three large categories: (a) math exercises needed to fix the notions and to develop mathematical skills, (b) concrete attacks on “small” implementations of the crypto systems and crypto protocols covered in class, and (c) exercises in which students are asked to analyze variations of crypto systems and crypto protocols and to reveal the weaknesses of the proposed variations. The textbook is a good source for exercises in the categories (a) and (c).

The list of suggested projects takes into account that the course audience includes students that have little or zero programming experience. Thus the list includes: (a) several programming projects (such as the implementation of differential attack on a small version of DES, RSA, different signature schemes, etc.), (b) projects that involve reading recent research articles and writing a survey paper, and (c) projects that ask the student to design protocols for some cryptography functionality using concrete “real-world” objects such as boxes with different kind of locks, pebbles, etc. Such protocols are used in cryptography as physical metaphors for digital implementations of the functionality and serve as a first intuitive step in the design of the protocol. One example of such a project is the following. Alice has a number n_a and Bob has a number n_b, both in the set of integers ranging from 1 to 100. They want to know which one has a larger number but they do not want to reveal any other information (or as little information as possible). They can use boxes and pebbles in their protocol. The pebbles are identical (they produce the same sound when the boxes are shaken). Your task is to design such a protocol and to analyze it. In your analysis, consider the number of pebbles, the number of boxes, and how much information is leaked. Ideally, if Alice learns that, say, n_b is larger than n_a, then, from her point of view, all the numbers larger than n_a should be equally likely to be n_b. For example consider that $n_a = 20$ and that at the end of the protocol, Alice knows that n_b is larger than *20*, that is n_b is one of the numbers *21, 22, ..., 100*. Then ideally $Prob(n_b = 21) = Prob(n_b = 22) = \ldots = Prob(n_b = 100)$. Solutions in which the probabilities are not equal but are close are acceptable (let's say the difference of any two probabilities in absolute value is at most *0.01*).

3.2 Network Security

This course covers the principles of network security, focusing on specific application areas such as authentication (Kerberos and X.509 certificates), email security (PGP and S/MIME), IP security (IPSec), transport layer security (TLS/SSL), and firewalls. The course begins with a general overview of common attacks and security mechanisms and services for attack prevention and detection, followed by a two-week introduction to basic cryptography. The required textbook is Network Security Essentials by William Stallings [19]. Students working in groups are responsible for a paper/presentation and completion of 3-4 assignments. The group size depends on enrollments; there are typically 3-4 members per group. In determining the course grade, the four components midterm, final, paper/presentation and assignments have equal weight. The course has been offered 6 times from Spring 2003 through Fall 2005, once as an independent study, and has seen a maximum enrollment of 10 students.

The material covered in the course is fairly detailed and the objective is to provide students with an understanding of the various methodologies and security protocols employed in network security. Exam questions have included a combination of short answer, multiple choice and problems.

The instructor assigns a topic for the paper to each group. Topics have included wireless LAN (802.11) security, cellular network security, denial of service attacks, and IP, ICMP and TCP vulnerabilities. The paper is written in a formal conference style requiring 4-5 pages in two-column format. While the paper is not expected to be overly technical, it is to be written at about the same level of detail as topics covered in the course text. The paper also provides an opportunity to assign topics in network security that are of current interest and address new developments in the field.

The assignments constitute the most interesting part of the course. For their first assignment, students use a socket program to implement the Diffie-Hellman exchange and transfer a message over a network encrypted using AES. They could write their own code, or use prewritten free software and tools available on the Web. Other assignments include using GPG and its trust model to transmit secure email, and using Snort to log packets, write rules and trigger alerts based on network protocols and message characteristics.

When the course was originally proposed, SNMP security was included as a topic. However, since students would need some background in network management and SNMP, it was decided to drop this topic due to a lack of time. The system security aspects of the course are limited to a discussion of firewalls from a network viewpoint. The reason is that topics such as viruses and worms, intrusion detection and honeynets are covered in other courses in the track. Both SNMP security and system security topics could be assigned as topics for the paper.

Network security has often been taught as a combined undergraduate/graduate course. This impacted the undergraduate course in that the material had to be presented at a somewhat higher level with a little more emphasis on the technical details. The differences in the level and background of the graduate and undergraduate students also created some difficulties. While the topics covered during lectures were the same for both courses, the graduate students were assigned additional papers for independent study and a semester project that was significantly more complex and required a substantial amount of programming. Presently, the undergraduate course is not being combined with the graduate course.

An issue that needs further discussion is that of prerequisites for the network security course. At first, a course in cryptography and a course in computer networks were prerequisites. However, since an overview of cryptography is presented at the beginning of the network security course, it was felt that the cryptography prerequisite may be omitted. Currently, only a course in computer networks is required.

Overall, the course has been quite successful judging from end-of-semester student evaluations. Assignments that address IP/TCP vulnerabilities, IPSec, SSL/TLS and wireless security would be beneficial. Unfortunately, we have not yet found assignments based on these topics that are non-trivial and possible to complete in approximately two weeks. We would also like to include more assignments that involve programming and require students to understand code that implement network security protocols.

3.3 Application Software Security

The Application Software Security course introduces students to the security concepts in developing software applications. This course discusses design principles for secure software development, and some of the security issues in current programming and scripting languages, database systems and web servers. This course is the only one that has been taught by more that one instructor. Three instructors with approaches varying from small number of hands-on projects and strong current literature research projects, to equal time allocated to lab and lecture, to completely lab-based, starting with the overview of Assembly language and run-time environment, have taught the course. The completely lab-based approach is being experimented during this semester (Fall 05), though we don't have the final data, from the preliminary feedback of the students, they all enjoy the course greatly. We do firmly believe that like other courses in this track, a strong lab component is necessary. Some of the topics and projects covered in this class include:

– Buffer overflow – Students are presented with an in-depth discussion of stack and heap overflow problems and how to exploits are generated. The reading assignments include papers addressing the overflow problem, integer and format string overflow problems, methods of defense against buffer overflow and secure programming techniques to prevent buffer overflow. In the lab, students work with relatively simple programs with buffer overflow vulnerability and asked to write the exploits. Though our computer science majors do take a course in Assembly language, we found that our undergraduate students don't have the needed skills to actually handle more challenging projects in this area.

– Threat modeling – There is a good coverage of this topic in "writing secure code" [12]. Microsoft has developed a tool [2] that is freely available and can be used to make the projects on this topic more interesting and challenging. Threat Modeling [20] provides a good reference for this project.

– Authentication and authorization – we provide students with a broad overview of authentication, authorization and access control techniques. Java security model is presented here and students' projects deal with using JAAS. Also we discussed Kerberos in details and a simple project dealing with configuring a Kerberos server in our security lab and using it to authenticate users for their projects.

– Cross-site scripting and SQL injection – Students enjoyed the discussion of both topics. Students did team projects demonstrating these problems.

3.4 Operating Systems Securit*y*

The operating systems security course introduces students to operating systems security issues and how to secure a system, and it has Operating Systems as a prerequisite. When the course was originally designed, we allocated equal time to Unix and Windows systems. However, based on our experience, we spend more time discussing Unix system than Windows. First, we realized that our students are less familiar with the Unix environment than Windows. Thus, we had to spend few weeks at the beginning to discuss the Unix operating system. Second, there are much better textbooks, articles and resources available addressing Unix security vulnerabilities than those for Windows. We tried to cover the same topics for both systems, to reinforce the concepts and allow students to make a better comparison of the features.

The coursework for this class consists of literature review, projects and exams. The students were assigned a paper each week to read and had to write a one-page review. Examples include "Security Report: Windows vs. Linux" [16], "Root Kits – An Operating Systems Viewpoint" [13], and "Leave no Trace, playing Hide and Seek Unix Style" [4]. Students very much enjoyed reading and reporting on these papers.

For the exercises and projects in this class, students had root or administrator access to their virtual machine. There are 7-8 projects small team projects in this class in addition to the final term project that depends on its scope, and can be either an individual or a team project. The projects dealing with Unix environment include writing a password cracker program; creating and managing user accounts; installing, configuring and using Tripwire [2]; configuring a simple secure ftp site and creating "jails"; and hardening Linux project using Bastille. We also used the NSA Security Configuration Guides for Windows 2000 [1] and found them very helpful and complete.

3.5 Case Studies in Computer Security

The Case Studies in Computer Security course serves as the capstone experience for the security track. Only offered in the spring semester, it has Operating Systems Security and Network Security as prerequisites. This course has been offered twice, in Spring 2004 and Spring 2005 to five and nine undergraduate students. It is a hands-on course that emphasizes defense, detection, and administration, and is arranged around a sequence of five or six competitive team-based laboratory exercises in the security laboratory. In a typical exercise, four different teams of students design and construct their own network of machines. They can choose from a range of operating systems, but their resulting network is required to provide a suite of remote services like Web, SSH, or FTP to their competing teams in the laboratory. Each team is then provided with authentication credentials to one or more of the services offered by some of the other teams, with the conditions that

- No team has root equivalent credentials on any machine from another team,
- No team has all of the non-root credentials for any other team, and
- No team knows which other teams have credentials for the services that they are to provide.
-

During the live portion of the exercise, each team must verify that the services provided by opposing teams for which they have authentication credentials are correctly functioning. They then try to illicitly gain access to all of the remaining services and the remote host itself, if possible. Once the live portion of the exercise has completed, students review their logs to try to determine who accessed their network, and whether they did so legitimately or illegitimately.

The purpose of the course is to give our graduates experience configuring and operating a network of machines in an unknown environment. Although attacks, and attack tools are described, the majority of the class time is spent learning defensive skills. Topics covered in the class include:

– Account and password management. PAM, password cracking.

– Logging and Auditing. Setting up a log server.

– Simple reconnaissance techniques; ping, nmap.

– Packet sniffers; Ethereal.

– Intrusion detection systems; Snort.

– Configuring common services: IIS, Apache, OpenSSH, WU-FTP.

– Advanced reconnaissance: Null connections and NetBIOS enumeration, SNMP walking.

– Backdoors: netcat, vnc.

– Firewalls. Iptables.

– Security analysis tools: Nessus, Microsoft baseline security analyzer.

– Security configuration tools: Bastille, Microsoft IIS lockdown tool.

For attacks, we present attack tools like Metasploit and the sniffer/password cracker Cain & Abel. The decision to give less weight to attacks and more weight to defense is not motivated by philosophical reasons, but rather by practical considerations. There is simply not enough time in the course to cover everything, and the purpose of the course is to teach potential security officers and system administrators- not penetration testers. For example, we have discovered as the class has been taught that it takes students a great deal of time- three or four weeks- to learn how to write good firewall rulesets, especially when they are to govern a realistic network with some complexity.

As an example of our live exercises, let us briefly describe the course's final exercise. In it, each of the four teams is told that they are the IT department for a hypothetical company. The company has offices in three different physical locations and they need to be able to work collaboratively on projects. The company also needs simple web presence, and the ability to deliver documents to the public. Finally the company has a corporate partner that should have access to additional information not available to the public, as well as the ability to work collaboratively on projects. The role of the partner company will be played by one of the other three teams. With these general guidelines, student teams are free to construct their own network and choose what services they will provide to meet these business requirements. Students begin by designing their network; in addition to setting up workstations to represent each office, they set up a collection of servers to provide the company's public presence. The students also set up and configure one or more logging servers, firewalls, and intrusion detection systems. All together, the preparation takes around two weeks. In the live portion of the exercise students access the services provided by both their partners and their competitors. Once the live portion of the exercise concludes, students write a report where they describe what they did to their competitors, and more importantly what their competitors did to them. Their project grade is primarily based on how well they set up their network, and how well they detect the attacks that other teams send their way.

4. ADMINISTRATIVE CHALLENGES

Like all new tracks, our security track has had its share of administrative challenges. Scheduling courses needed for completion of the track has presented some difficulties. Currently, we are offering courses according to the following plan:

– Fall: Cryptography, Network Security, Operating System Security

– Spring: Application Software Security, Case Studies

In the past, however, Operating Systems Security, Application Software Security and Network Security have been offered in both Fall and Spring. This has been due to several reasons including newness of the track, prerequisites, student demand, expediting graduation, and faculty availability.

Another factor that has impacted scheduling in the past is the offering of combined undergraduate and graduate classes. During the first few semesters the track was offered, some classes had few students. This was to be expected until such time that new students in the track would become juniors or seniors and start taking the security courses. In fact, at the beginning, many students taking these classes were not in the security track. Some were not even computer science majors. This together with the combined undergraduate and graduate enrollments ensured that class sizes were administratively acceptable. Now that the track has been available for a few years and student awareness of the benefits of completing the track has increased, we expect reasonably steady enrollments in all courses.

At present, there are only five faculty members teaching the security courses. All are tenured or tenure-track; one has a joint appointment in Mathematics and Computer Science; the others are in Computer Science. With the exception of Application Software Security, the same faculty member has always taught every offering of a given course. So far, this has worked out well, since the courses have matched the interests and expertise of the faculty. It also allowed courses to be fine-tuned and improved with each offering, and provided continuity. However, this approach presents difficulties in that faculty availability and track schedules may not always match. Although the department currently has about 24 tenured/tenure-track faculty members, most have interests in other areas of Computer Science and Information Systems, which means there is little flexibility in assigning instructors to security courses.

5. FUTURE PLANS

As we enter the fourth year of the track, we have begun a complete review of the track and its courses. Currently, we are considering a number of modifications and improvements to the track. They include:

– Changing the prerequisite tree for the track. In particular, we are considering removing the prerequisite course in Cryptography from the Network Security course; this will reduce the size of the (lengthy) chain of courses that our students need to graduate. As it stands, Cryptography is a prerequisite for Network Security, which is a prerequisite for Case Studies in Computer Security; this long chain of 300 and 400 level courses has been a difficulty for students.

– We are assessing the contribution of the introductory course, Introduction to Information Systems Security, to the whole track. This course provides a broad overview of technical and human components of information systems security. Therefore, there is a considerable overlap and redundancy between this course and the remaining courses in the track We are considering the possibility of reusing the credit hours to design a new course.

– We may want to designate one of the sections of computer ethics course for the security track students to specifically address some of the security policies and regulations. Right now, the Computer Ethics course is a general course for all of our majors, and we may want to use this course to emphasize the ethical issues that arise in computer security.

– We are trying to forge closer connections between the different courses. Some topics occur naturally in different courses; for example firewalls and intrusion detection systems are described in both the Network Security course and the Case Studies course. We would are trying to ensure that the topic is covered in a complementary fashion in the two courses.

Another issue that has arisen is the rapid pace of change in the field. Given the evolving nature of computer security, to what extent do course assignments and projects need to reflect ongoing developments? For example, the weaknesses of 802.11 WEP are well known. Should students be given a hands-on assignment using WEP and the interim WPA? What elements of the proposed 802.11i security standard need to be included? Should the topic of wireless security be assigned as a paper topic in the network security class, or should it be included as one of the topics covered in the lectures?

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under grant No. DUE-0113783

REFERENCES

1. NSA Security Configuration Guides for Windows 2000. Internet http://www.nsa.gov/snac/index.cfm
2. Threat Modeling Tool. Internet http://www.microsoft.com/downloads/details.aspx?familyid=62830f95-0e61-4f87-88a6-e7c663444ac1&displaylang=en. Accessed September, 2005.
3. Tripwire. Internet http://sourceforge.net/projects/tripwire/ Accessed September 2005.
4. Accident, P. Playing Hide and Seek, Unix style. Phrack Magazine, Issue 43, File 14. Internet: http://www.phrack.org/phrack/43/P43-14 Accessed September 2005.
5. Azadegan, S., Lavine, M., O'Leary, M., Wijesinha, A., and Zimand, M. 2003. A dedicated undergraduate track in computer security education. In Security Education and Critical infrastructures, C. Irvine and H. Armstrong, Eds. Kluwer Academic Publishers, Norwell, MA, 319-331.
6. Azadegan, S., Lavine, M., O'Leary, M., Wijesinha, A., and Zimand, M. 2003. An undergraduate track in computer security. In Proceedings of the 8th Annual Conference on innovation and Technology in Computer Science Education (Thessaloniki, Greece, June 30 - July 02, 2003). D. Finkel, Ed. ITiCSE '03. ACM Press, New York, NY, 207-210.
7. Bogolea, B. and Wijekumar, K. Information security curriculum creation: a case study. In InfoSecCD '04: Proceedings of the 1st annual conference on Information security curriculum development, pages 59--65, New York, NY, USA, 2004. ACM Press.
8. Dornseif, M., Freiling, F. (geb. Gärtner), Holz, T. and Mink, M. An offensive approach to teaching information security: "Aachen summer school applied it security". Technical Report AIB-2005-02, Aachen, 2005.
9. Dornseif, M., Gaertner, F.C., Mink, M. and Pimenidis, L. Teaching data security at university degree level. Proceedings of the WISE04 (to appear). Internet: http://md.hudora.de/publications/.
10. Fulp, J. The bastion network project. In C. Irvine and M. Rose, editors, Avoiding Fear, Uncertainty and Doubt Through Effective Security Education, pages 65--71. Center for Information Systems Security Studies and Research, Naval Postgraduate School, 2004.

11. Hill, J.M.D., Curtis, J., Carver, A., Humphries, J.W., and Pooch, U.W. Using an isolated network laboratory to teach advanced networks and security. In SIGCSE '01: Proceedings of the thirty-second SIGCSE technical symposium on Computer Science Education, pages 36--40, New York, NY, USA, 2001. ACM Press.
12. Howard, M. and LeBlanc, D. Writing Secure Code, Microsoft Press, 2003.
13. Kühnhauser, W. E. 2004. Root Kits: an operating systems viewpoint. SIGOPS Oper. Syst. Rev. 38, 1 (Jan. 2004), 12-23.
14. Lathrop, S.D., Conti, G.J., and Ragsdale, D.J. 2003. Information warfare in the trenches. In Security Education and Critical infrastructures, C. Irvine and H. Armstrong, Eds. Kluwer Academic Publishers, Norwell, MA, 19-39.
15. Perrone, L. F., Aburdene, M., and Meng, X. Approaches to undergraduate instruction in computer security. Proceedings of the American Society for Engineering Education Annual Conference and Exhibition, ASEE 2005. Internet: http://www.ists.dartmouth.edu/library/116.pdf.
16. Petreley, N. Security Report: Windows vs Linux, The Register. Internet http://www.theregister.co.uk/security/security_report_windows_vs_linux/ Accessed September 2005.
17. Romney, G.W. and Stevenson, B.R. An isolated, multi-platform network sandbox for teaching it security system engineers. In CITC5 '04: Proceedings of the 5th conference on Information technology education, pages 19--23, New York, NY, USA, 2004. ACM Press.
18. Schafer, J., Ragsdale, D.J., Surdu, J.R., and Carver, C.A. The IWAR range: a laboratory for undergraduate information assurance education, Proceedings of the sixth annual CCSC northeastern conference on The journal of computing in small colleges, p.223-232, April 2001, Middlebury, Vermont, United States.
19. Stallings, W. Network Security Essentials: Applications and Standards. Prentice Hall, 2nd Edition, 2003.
20. Swiderski, F. and Snyder, W. Threat Modeling. Microsoft Press, 2004.
21. Tikekar, R. and Bacon, T. The challenges of designing lab exercises for a curriculum in computer security. J. Comput. Small Coll., **18**(5):175--183, 2003.
22. Tikekar., R.V. Teaching computer security to undergraduates. In C. Irvine and M. Rose, editors, Avoiding Fear, Uncertainty and Doubt Through Effective Security Education}, pages 5--10. Center for Information Systems Security Studies and Research, Naval Postgraduate School, 2004.
23. Trappe, W., and Washington L. Introduction to Cryptography with Coding Theory, Prentice Hall, 2006 (second edition).
24. Vigna, G. Teaching Hands-On Network Security: Testbeds and Live Exercises. Journal of Information Warfare, **3**(2):8--25, 2003.
25. Vigna, G. Teaching Network Security Through Live Exercises. In C. Irvine and H. Armstrong, editors, Proceedings of the Third Annual World Conference on Information Security Education (WISE 3), pages 3--18, Monterey, CA, June 2003. Kluwer Academic Publishers.
26. Wagner, P.J. and Wudi, J.M. Designing and implementing a cyberwar laboratory exercise for a computer security course. In Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education (Norfolk, Virginia, USA, March 03 - 07, 2004). SIGCSE '04. ACM Press, New York, NY, 402-406.
27. Yang, T. A., Yue, K., Liaw, M., Collins, G., Venkatraman, J. T., Achar, S., Sadasivam, K., and Chen, P. Design of a distributed computer security lab. J. Comput. Small Coll. 20, 1 (Oct. 2004), 332-346.

INFORMATION SECURITY ATTACK TREE MODELING

An Effective Approach for Enhancing Student Learning

Jide B. Odubiyi and Casey W. O'Brien

Department of Computer Science, Bowie State University, Bowie, MD – jodubiyi@cs.bowiestate.edu

Network Technology Department, Community College of Baltimore County, Baltimore, MD – cobrien@ccbcmd.edu

Abstract: This paper presents a framework for enhancing student learning about the vulnerabilities of information assets of a business enterprise using attack tree modeling. Using this framework, students get an overview of the methodology as well as learn how to implement it with a well-known list of information security vulnerabilities. As a result, students can provide input into threat modeling strategies and operating procedures and thus, increase overall confidentiality, integrity, and availability of computer and network systems. The paper concludes by describing a corresponding system capability metric that a system administrator, student, or red team can use to test the vulnerabilities of the systems within a business enterprise.

Keywords: Attack Trees, Threat Modeling, Intrusion Scenarios, Attack Scenarios, Business Enterprise, System Capability Metric, Failure Mode and Effect Analysis

1. INTRODUCTION

For years, engineers have relied on the analysis of failure data to improve their designs by employing the engineering principle of Failure Mode and Effect Analysis (FMEA). The main goal of this principle is to identify risks and initiate concerted efforts to control or minimize these risks. By knowing the risks, the project plan can be created more realistically.

FMEA techniques [1] help to identify failure potential in a design or process before it actually happens. This is done by following a procedure which helps to identify design features or process operations that could fail. This procedure requires the identification of all potential failures, including the causes and the effects of those failures. Using knowledge of the design, process, and/or system, estimates of the probability of these failures can be identified, together with the severity of the effects of the failure, and the probability of detecting the failure before it becomes critical.

While engineers have historically been good at using failure data to improve their designs and systems, software engineers and information systems administrators have not. In general, IT professionals dealing with the security of systems do not use similar failure data - attack data - to improve their computer and network systems and the related components that are a part of those systems. The reasons for this vary from organizations that are weary of divulging such information (for fear of decreased public confidence), to fears that attackers will employ the attacks against their systems, to a lack of detailed and reliable data on attacks [2].

Despite weariness on the part of organizations to disclose attacks on their systems, attack data have become more available in recent years, due in large part to the public and media's increased attention on information security issues. In addition to the public and media interest in information security-related issues, other resources like the SANS Internet Storm Center [3] and Security Focus [4], as well as many Computer Emergency Readiness Teams (CERTs) and Coordination Centers (CCs) throughout the world bring together timely and comprehensive security-related information. These resources serve as a forum for in-depth discussions and announcements of computer and network security vulnerabilities. They detail what they are, how to exploit them, and how to fix them.

Countless articles and research publications have been written detailing threat models that can be used to test the security of an enterprise system. The authors of these publications argue that understanding all of the different ways a system can be attacked can help IT professionals responsible for securing systems design and implement countermeasures, to thwart those attacks. In addition, if these same IT professionals can understand who the attackers are and what their abilities, motivations, and goals are, they can implement the proper countermeasures to deal with the real threats [5].

A fundamental understanding of threat modeling can help system developers/integrators build robust and reliable systems. If they know the possible threats, they can use an attack tree to test their systems. For example, after building a Web site, a system administrator can readily apply a threat model to test it. The same is true of students learning about the vulnerabilities of the information assets of a business enterprise. By developing several attack trees (which constitute an attack forest of a business enterprise), students get an appreciation for the hundreds of possible vulnerabilities in a given enterprise system, and the challenges faced by system administrators when securing their computer and network systems. Ultimately, students would then use this model to further test the security of their systems. Several organizations, including The General Accounting Office and the U.S. Department of Homeland Security, employ Red Teams to identify vulnerabilities in their information assets [6].

2. TEACHING THREAT MODELING

Students cannot build and test the security of computer and network systems unless they understand the threats posed to those systems. Therefore, it's critical that students in Information Assurance (IA), Computer Science, network technology, and other IT related programs be taught not only how to design and build secure systems, but also how to identify the means, motives, and opportunities of their adversaries, identify the threats posed to the systems they are securing, and ways to test it.

When the above mentioned academic programs are infused with threat modeling concepts and implementation strategies for testing the security of systems, students can do more. They can extend the application of this framework to design systems that meet the security objectives of an organization, decide on trade-offs during key design decisions, and help reduce the risks of security-related issues arising during the implementation and operations phases.

3. THE CLASSROOM ENTERPRISE

The following is an example of how we introduced and enhanced student's understanding of threat modeling. We asked students to compile a list of information assets managed by the university or a typical medium business enterprise. The list of information assets includes several Web servers and Database Management Systems for Payroll, Strategic Plans, etc. We then provided the students with 32 information system vulnerabilities including the Twenty Most

Critical Internet Security Vulnerabilities [11] and asked them to develop attack trees for them. By developing several attack trees which constitute an attack forest of a business enterprise, students moved from a conceptual grasp of threat modeling, to an appreciation for the hundreds of possible threats to a system, to the challenges faced by system administrators, to methodologies for testing the security of a system.

3.1 The Adversary Model

Before students are able to design, build, and implement secure systems, they must understand the means, motives, and opportunities of their adversaries [5, 9]. Salter, et al [9] developed an Adversary Model to help organizations thoroughly understand their adversaries and characteristics. This model characterizes adversaries in terms of their resources, access, risk tolerance, and objectives. The learning objective for students is that countermeasures are only needed for attacks that meet the adversaries' resources and objectives, but to be useful, the countermeasures must also meet the organization's needs for cost, ease of use, compatibility, performance, and availability.

3.2 Identifying and Classifying Assets

The task of identifying assets that need to be protected within an organization is one of the less glamorous aspects of information assurance. Unless an organization know its assets, where they are located, and what their value is, it's difficult to decide on the amount of time, effort, and money that should be spent on securing them. In addition, organizations won't know what the adversaries of their systems want until they've identified the sensitive information and related assets on those systems. Students should explore asset classification models, which can be categorized as follows: the identification of an organization's assets, the accountability of those assets, preparing a framework for classifying those assets, and implementing the classification framework.

3.2.1 The Enhanced Telecom Operations Map for the Telecom Service Industry

Figure 1 can serve as an excellent resource to give the learner an appreciation for the processes required for managing and protecting a telecommunication business enterprise. The enhanced Telecom Operations Map (eTOM) has been developed by The Telecom Management Forum as a generic framework for organizing a telecommunication business organization into three major areas. The three areas consist of (a) Planning and Lifecycle Management of the Strategy, Infrastructure, and Product development; (b) Operations Management, and (c) Enterprise Management—Business Support management. Each of the three areas consists of computer systems, telecommunication networks and software that must be protected. The responsibility for protecting these assets falls on the shoulder of the chief security officer or the Information Technology (IT) department. As a result of mergers and acquisitions, there can be tens of software products just for the Billing operation. At one time, the former WorldCom had over 30 billing systems. Each billing system consisted of database management systems, Web servers, and accounting software. Other operations support systems (OSSs) consist of diverse software products. The eTOM framework provides an environment for students to identify OSSs to analyze for vulnerabilities.

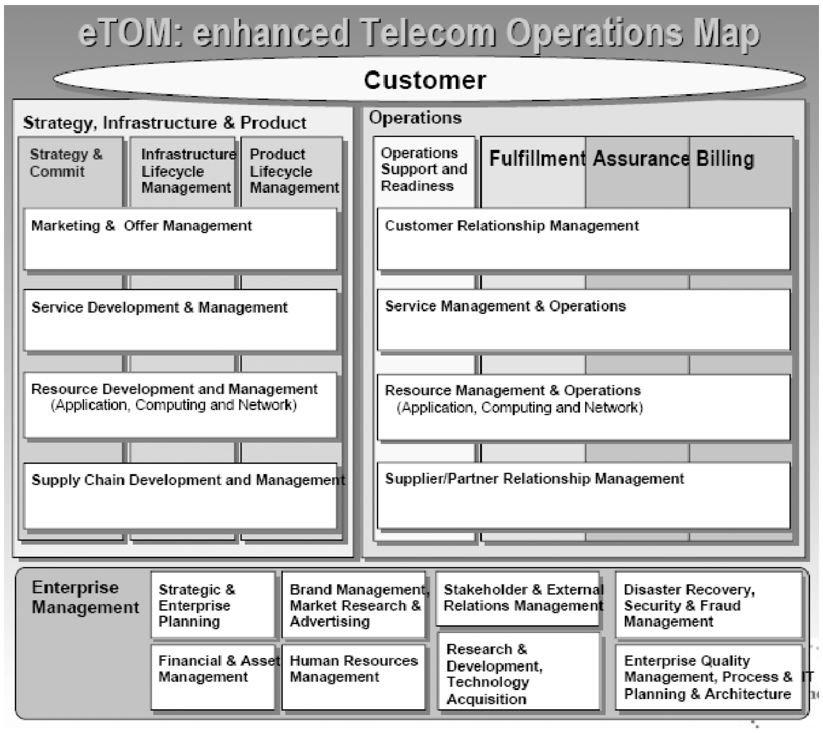

eTOM: enhanced Telecom Operations Map
Customer
Strategy, Infrastructure & Product
Strategy & Commit
Infrastructure Lifecycle Management
Product Lifecycle Management
Marketing & Offer Management
Service Development & Management
Resource Development and Management (Application, Computing and Network)
Supply Chain Development and Management
Operations
Operations Support and Readiness
Fulfillment
Assurance
Billing
Customer Relationship Management
Service Management & Operations
Resource Management & Operations (Application, Computing and Network)
Supplier/Partner Relationship Management
Enterprise Management
Strategic & Enterprise Planning
Brand Management, Market Research & Advertising
Stakeholder & External Relations Management
Disaster Recovery, Security & Fraud Management
Financial & Asset Management
Human Resources Management
Research & Development, Technology Acquisition
Enterprise Quality Management, Process & IT Planning & Architecture

Figure 1. The Enhanced Telecom Operations Map® (eTom) (Source: The Telemanagement Forum) [15]

3.3 Creating an Architectural Overview

When students are creating an architectural overview of an enterprise system, they need to be explicit about what the network and related systems are designed to do, how they plan on engineering it to achieve that functionality, and what technologies are required to implement the design. This helps them identify the common technology-specific threats and implement solutions to overcome them [7]. In addition, having students create a diagram of how information flows throughout the organization allows them to discover the critical components and procedures of the enterprise system. The student can use the eTOM model in Figure 1 or the instructor may identify a similar map for another industry.

3.4 The Sum of Its Parts

Once students have an overview of the architecture of the enterprise, they can use it to break down the system into its constituent components. The more an organization knows about its system, the easier it is to discover threats against it. The steps involved in breaking down a system into its constituent parts starts with creating a security profile, that is, an analysis of the system's strengths and weaknesses. Students will need to validate all data being sent across their systems. This includes a comprehensive examination of the trust boundaries of the enterprise network, the flow of data in and out of the enterprise, and any entry points into the system that will ensure that the flow of information is done in a secure manner [8].

3.5 Identifying Threats Using Attack Trees

The first step is to identify the threats that might affect an organization's system and potentially compromise its assets. This includes identifying network, host, and application threats. Network threats can be assessed by having students investigate how the data passes through routers, firewalls, switches, and other network devices. Students need to understand the logic and syntax of these device's configuration files. In addition, students need to be able to determine what it takes to get past or compromise each device. Host investigations should include common configuration categories applicable to all server and operating system resources (patches, files/directories, ACLs). Finally, students should examine the enterprise's application software [7].

There are other ways students can identify the threats to a given enterprise system, namely using a list of known threats [11] and using attack trees with attack patterns. We encourage using a combination of both to identify threats to a system. When only using a previously prepared list of known threats, the resulting threat list only reveals the common, known threats. Using attack trees, in conjunction with a list of known threats, can help identify other potential threats.

3.6 Attack Trees and the SANS Top-20 Vulnerabilities

An attack tree provides a method for representing attacks (and similar vulnerabilities) on a system in the structure of a tree. The goal of the tree is the root node. The leaf nodes represent different paths to achieve the goal. As depicted in Figure 1, a business enterprise typically consists of hundreds of information assets that are vulnerable to attacks. Each of these assets can be modeled as an attack tree resulting in an attack forest [5]. The root of each tree in the attack forest constitutes an event that can potentially damage the enterprise system and its related resource's confidentiality, integrity, and availability. The SANS twenty most critical Internet security vulnerabilities provide a rich environment for the students to learn about well documented attacks [11]. The SANS top-20 2004 vulnerabilities consist of two top-10 vulnerable services in Windows and UNIX operating systems. The SANS top-20 list is updated annually and provides useful information about each vulnerability, systems affected, and how to protect the services. Students can use the protection measures to develop attack scenarios.

3.6.1 The Mail Transport Service and Enterprise Service (NIS/NFS) Vulnerabilities

Two of the twenty critical vulnerabilities identified by SANS are the Mail Transport Service (MTS) and the misconfigured enterprise services: Network File System (NFS) and Network Information Service (NIS).

The Simple Mail Transfer Protocol (SMTP) is one of the oldest Internet application protocols that employ Mail Transport Agents (MTAs) as servers to send emails from senders to recipients. Examples of these MTAs are sendmail, QMail, Courier-MTA, Postfix, and Exim. They are vulnerable to buffer/heap overflow attacks when they are not patched or when their patches are out of date. An attacker can compromise the Mail Transport Service by exploiting open relays (i.e., a situation that permits the sender and receiver that are not part of the domain to send and receive mail). A third major vulnerability of the MTS is exploitation of non-relay problems such as misconfiguration of the user account database, thereby exposing the service to spam attacks.

The NFS and NIS services are used in UNIX servers to hold directories of files systems and to provide locations of distributed databases respectively. They are vulnerable to buffer overflow attacks due to unpatched services, Denial of Service (DoS) attacks and weak authentication. Given the goal of attacking either the MTA or the three sub-goals listed above, the student can be assigned to develop attack scenarios for the services using the approach described in the next paragraph.

3.7 Modeling an Attack Tree

The Web server is one of the SANS top-20 most critical vulnerabilities due to add-on software modules such as CGI scripts, PHP bugs, servelets, etc. The process for modeling an attack tree for a Web server's vulnerability is described in this paragraph. Since each tree has a root node that represents the attacker's goal, and the leaf nodes represent different paths to the root, each child node represents the steps an attacker can take. Modeling the attack tree involves associating a logical AND and a logical OR with each node. In essence, a node of an attack tree can be decomposed into an AND or an OR node. An AND node or an OR node decomposition can be represented in graphical or textual formats. Figure 2 is a textual representation of one of the top vulnerabilities to Windows-based systems [11]:

GOAL: (G0) Gain Privileged Access to a Web Server Using a Known Vulnerability [8]

AND G1. Identify organization's domain name.

G2. Identify organization's firewall IP address

OR 1. Interrogate domain name server

2. Scan for firewall identification

3. Trace route through firewall to Web server

G3. Determine organization's firewall access control

OR 1. Search for specific default listening ports

2. Scan ports broadly for any listening port

G4. Identify organization's Web server operating system and type

OR 1. Scan OS services' banners for OS identification

2. Probe TCP/IP stack for OS characteristic information

G5. Exploit organization's Web server vulnerabilities

OR 1. Access sensitive shared intranet resources directly

2. Access sensitive data from privileged account on Web server

Figure 2. Web Server Attack Description

As presented in Figure 2 above and stated earlier, a node of an attack tree can be decomposed into an AND or an OR node. Both the AND and the OR decompositions can be represented in graphical or textual format as shown in Figures 3 and 4 below. All the attack sub-goals (such as G1, G2, G3, …., G5) must be achieved for the exploit to succeed.

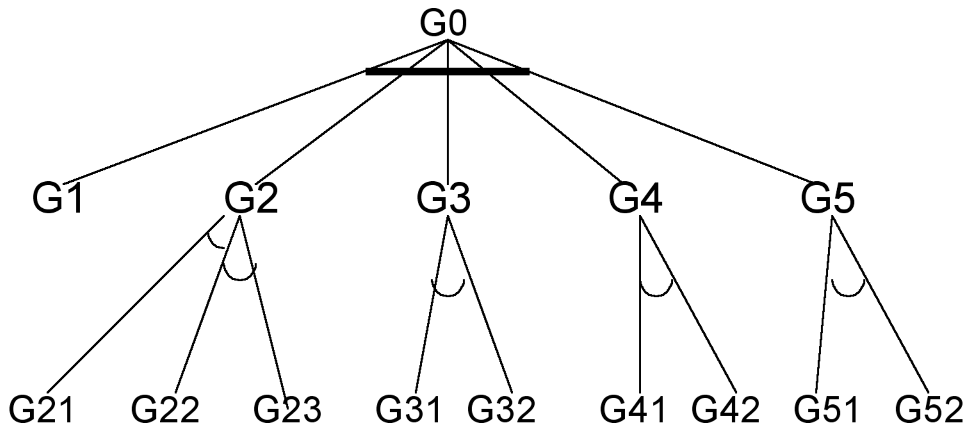


Figure 3. A Graphical Representation of an Attack Tree described in Figure 2
(G0 ≡ G1 ∩ G2 ∩ G3 ∩ G4 ∩ G5); (G2 ≡ G21 ‖ G22 ‖ G23)

The attack tree in Figure 3 can generate 24 attack or intrusion scenarios [10] in Figure 4 below. The 24 possible attack scenarios are:

[G1, G21, G31, G41, G51], [G1, G21, G32, G41, G51],

[G1, G21, G31, G42, G51], [G1, G21, G32, G42, G51],

[G1, G22, G31, G41, G51], [G1, G22, G32, G41, G51],

[G1, G22, G31, G42, G51], [G1, G22, G32, G42, G51],

[G1, G23, G31, G41, G51], [G1, G23, G32, G41, G51],

[G1, G23, G31, G42, G51], [G1, G23, G32, G42, G51]

[G1, G21, G31, G41, G52], [G1, G21, G32, G41, G52],

[G1, G21, G31, G42, G52], [G1, G21, G32, G42, G52],

[G1, G22, G31, G41, G52], [G1, G22, G32, G41, G52],

[G1, G22, G31, G42, G52], [G1, G22, G32, G42, G52],

[G1, G23, G31, G41, G52], [G1, G23, G32, G41, G52],

[G1, G23, G31, G42, G52], [G1, G23, G32, G42, G52]

Figure 4. Twenty-four possible attack scenarios

The realization that a simple Web server attack tree can generate 24 attack scenarios helps the student to understand the importance of developing secure software as a countermeasure against Web server exploits.

3.8 Benefits of Documenting the Threats: From an Attack Tree to an Attack Forest

Figure 3 is one of hundreds of vulnerabilities that make up an organization's attack forest. By creating an attack forest, an organization not only has a roadmap for testing for both known and unknown vulnerabilities, but also a means to document the threats to their systems and assets.

Ultimately, the attack forest document for an organization will include; each threat; a description of the threat; the target of the attack; the risk of the attack; the techniques likely to be used in carrying out the attack; and a risk management strategy. For example, when dealing with a SQL injection attack, the target is the database and the technique is that the attacker types a command into a textbox that is automatically added into a T-SQL command without client-side validation. To counter this threat, one can use regular expressions to validate the user name, and use a parameterized query to access the database [12].

4. CHALLENGES

Teaching threat modeling can be difficult for a number of reasons, each of which is described below.

4.1 From Past to Present

Organizations trying to protect their assets from attackers and current intrusion detection tools face a common problem: Being able to generalize from previously observed behavior to recognize future behavior, either malicious or normal.

Signature-based misuse detection techniques are acutely prone to this problem, as are anomaly-based detection tools that must determine normal behavior that is not identical to past observed behaviors, in order to reduce false alarms [13].

By understanding this problem, students can use both advances in intrusion detection techniques and adversary models as ways to better understand who their attackers are, and ultimately, implement the proper countermeasures to deal with the threats.

4.2 Input Data: Developing Attack Trees

As stated earlier, there are multiple ways students can identify the threats to a given enterprise system. We encourage using a combination of known threats [11] as the basis for developing an enterprise attack forest. When combining known lists of vulnerabilities with attack trees and attack trees' patterns, new vulnerabilities can be identified.

5. FUTURE WORK

If a simple attack tree (like the one outlined in Figure 3) can generate 24 attack scenarios and there are hundreds of attack trees in a medium-sized business, it is feasible to have thousands of possible attack scenarios. Most organizations do not have all the resources that enable their system administrators to exhaustively test all of the thousands of possible scenarios. We are working on developing a framework that will generate a sampling scheme that will provide a degree of confidence metric to aid systems administrators and Red Teams in selecting scenarios to test.

At the conclusion of their testing and using the sampling scheme, the testing team will be able to claim, with a degree of confidence, that 90% or 95% or 98% of all the attack scenarios have been tested. We have successfully demonstrated the approach to resolve aircraft to aircraft in-flight conflict scenarios and maneuver strategies for the Federal Aviation Administration (FAA) and in testing porosity of graphite composites at the Boeing Aircraft Company in Seattle, and we are optimistic of the potential success in the information assurance domain and its educational value in enhancing student learning.

REFERENCES

1. Shooman, M. L. Probabilistic Reliability: An Engineering Approach. McGraw-Hill Book Company, 1968.
2. Anderson, R., Why Cryptosystems Fail, Proceedings of the 1st ACM Conference on Computer and Communications Security, 1993.
3. SANS Internet Storm Center, http://isc.sans.org
4. Security Focus, http://www.securityfocus.org
5. Schneier, B., Attack Trees, *Dr. Dobb's Journal*, Vol. 24, no. 12, December, 1999, pp. 21-29.
6. Ray, H. T., Vemuri, R., & Kantubhukta, H. R. Toward Automated Attack Model for Red Teams, IEEE Security & Privacy, Vol. 3, No. 4, IEEE Computer Society, July/August 2005, pp. 18-24.
7 Howard, M. and LeBlanc, D., Writing Secure Code, 2nd Edition, Microsoft Press, 2002.
8. Meier, J.D., Mackman, A. & Wastell, B., Walkthrough: Creating a Threat Model for a Web Application. Retrieved August 27, 2005, from http://msdn.microsoft.com/security/securecode/threatmodeling/default.aspx?pull=/library/en-us/dnpag2/html/tmwawalkthrough.asp#tmwawalkthrough_breakingdowntheapplication
9. Salter, C., Saydjari, O.S., Schneier, B., & Wallner, J., Toward a Secure System Engineering Methodology, New Security Paradigms Workshop, September, 1998.
10. Moore, A. P., Ellison, R. J., & Linger, R. C. Attack Modeling for Information Security and Survivability. Technical Note: CMU/SEI-2001-TN-001., March 2001.
11. The SANS Institute, The Twenty Most Critical Internet Security Vulnerabilities (Updated) – The Experts Consensus, http://www.sans.org/top20/
12. Threat Model Your Security Risks, Microsoft Security Developer Center. Retrieved on August 27, 2005, from http://msdn.microsoft.com/security/securecode/threatmodeling/default.aspx?pull=/msdnmag/issues/03/11/resourcefile/default.aspx
13. Ghosh, A.K., Schwartzbard, A., A Study in Using Neural Networks for Anomaly and Misuse Detection, Proceedings of the 8th USENIX Security Symposium, August 23-26, 1999, Washington, D.C.
14. Mackman, A, Meier D.J.Meier,& Wastell B., Threat Modeling Web Applications. Retrieved July 29, 2005, from http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnpag2/html/tmwa.asp
15. The TeleManagement Forum www.tmforum.org

ADDING WHEN, WHERE, AND WHY TO HOW

Providing Ethical Context in Aggressive Information Security Labs

Scott J. Roberts and Andrew L. Reifers

School of Information Sciences and Technology, Penn State University – University Park,

NSA Center for Information Assurance Excellence

Abstract Educators recognize that laboratory based computer security courses do far more for students' understanding than purely theoretical courses. Laboratories with common hacker tools are invaluable for students in an Information Assurance curriculum. These tools help students better understand threats and the defense mechanisms needed to protect individual systems and entire networks from these attacks. Students greatly benefit from understanding the threats they are called to defend against, making them more effective protectors of enterprise or government networks. Conversely these labs offer students hands on experience with tools that could easily be turned and used against others. Through the use of situations that show ethical uses of these techniques students can have these beneficial experiences without becoming the very danger they are being trained to protect against.

Key Words Information Assurance, Ethics, Laboratory Exercises

1. INTRODUCTION

With the growth of networked businesses and organizations many positive elements of society have branched out and begun to flourish into cyberspace. Commerce has fully embraced the Internet, and e-commerce is becoming more mainstream everyday. Even personal journals, now taking the e-form blogs, have moved into this world wide public forum. With nearly all information and an ever-increasing amount of financial transactions taking place online societies negative elements are sure to follow. Crime goes where the money is and thus the Internet has quickly given birth to a new breed of criminals. Using new methods to accomplish old goals, this new breed of lawbreakers function in a way that has never been seen before, functioning across borders and legal jurisdictions, far outside the realm of traditional law enforcement. The Internet has, to a limited extent, become the new Wild West: lucrative, expanding, and largely lawless.

Many universities have joined the fight by beginning to train a new breed of sheriffs to protect this new e-territory. Like sheriffs of old the line between the criminals and the defenders of justice isn't always as clear as good versus evil. Doc Holliday and Wyatt Earp probably rode the same breed of horses and rode in the same style as Billy Clayton and his band of outlaws. They'd have shot the same sort of firearms, using the same technique, loaded with the same ammunition. Holliday and Earp would have probably been just as able, if not more so, to plan a jailbreak, hijack a stagecoach, or even rob a bank. [1] While this should not be taken as advocating vigilantism in computer security courses it illustrates an intriguing point. In most respects, from clothing to methods, the law enforcers were little different from the law breakers. Except for a small metal badge carried by the lawmen the only real difference between these very different groups was their motivation. The “how” of doing their job was nearly identical, but the “when”, “where”, and “why” was completely different. Many of the same “how’s” can be used as easily to protect as to attack ([2],[3],[4]).

In very much the same way the digital sheriffs being trained by today’s universities, technical schools, and certification programs are learning in the same way. Understanding how to lock down a firewall also shows a student how to cripple one. Teaching a student a tool that enumerates vulnerabilities to be patched also shows them how to enumerate vulnerable machines for exploitation. Although dangerous when misused, these tools are still of great use to an information security professional, when used in the right context. How could law enforcement investigate crimes without understanding how they are committed? How could a police officer keep people safe if he isn’t capable of firing his weapon at a dangerous criminal? In the same way that a police officer understands the correct “when”, “where”, and “why” to fire his weapon, Information Assurance professionals should know the when, and where to use of techniques and tools they possess. This becomes especially relevant when discussing the use of malicious tools.

2. CURRENT CURRICULUM

At Penn State University much work has already been done to provide the best possible learning experience for future Information Assurance Analysts, Engineers, and Managers. Inside the School of Information Science and Technology the Information Assurance program works to educate these future leaders in many aspects of computer security. All Information Assurance students are required to take the standard set of introductory IST courses, including an Introduction to Networking, an Introduction to Databases, and two basic programming courses, one taught in the School of IST and one from the Computer Science department. Once this background is established the Information Assurance specific courses become available. All Information Assurance Track students are required to take the introductory technical course, Network Security, as well as the introductory policy and ethics course, the Legal and Regulatory Environment of Privacy and Security. In addition each student is required to take one additional Information Assurance related course of their choosing, such as Wireless Network Design and Security, Web Application and Database Security, or in some cases an independent study with an Information Assurance track professor.

Laboratory requirements vary from class to class, though typically, as would be expected, the technical courses have a lab component. These laboratory times are conducted outside of class, and are generally dependent on the current lecture track, though this is not always the case. Students in teams of 5 to 8 people work together through a technical lab proctored by a Teacher’s Assistant (TA). These labs cover both malicious and defensive tools, giving students a basic overview of their setup and operation. Afterwards students are given a homework sheet with a few basic questions about the technology used and its operation.

The current Information Assurance lab curriculum includes two weeks studying the attack tools Metasploit Framework and L0pht Crack Password Cracker. The other eight weeks of lab detail use of a piece of standard Cisco network defense appliances and technology. Weeks three and four teach the use and maintenance of a Cisco router, weeks five and six are spent learning the Cisco Pix firewall, seven and eight are a Cisco Intrusion Detection System, and the last two weeks round out the series studying a Cisco Virtual Personal Network appliance. All these labs were Cisco approved lab sessions also taught by a graduate student TA.

3. HOLES IN ATTACK UNDERSTANDING

One of the biggest problems in any undergraduate Information Security/Assurance curriculum is balancing the many elements that need to be taught [5]. Computer security is a constantly evolving and changing world, defenders are being pressed to understand and deal with new threats, old threats revisited, and old threats subtly changed. At this point it becomes a struggle to keep up to date and decide what new issues need to be brought to the students attention and what can simply be left for them to discover once they reach their places of employment. On top of this is the difficulty of pacing a course correctly. Computer Security and Information Assurance cover a huge realm, from programming to policy, network administration to system configuration, and now spyware to spam [5]. It's difficult enough for most classes to get through the defensive basics and how to develop security policy. It is often impossible to even begin the most basic overview of attacks such as buffer overflows, configuration exploitation, spyware, spam, and the many other challenges facing individuals, corporations, and governments.

It's quite understandable that defensive technology and the policy of running security in a corporate security is the priority of academic network security courses. The vast majority of Information Assurance students will be working in defensive roles, protecting network infrastructures, not attacking other networks. With the exception of government run and funded Information Warfare, corporate penetration testing, and underground hacking competitions such as Root-Fu or Capture the Flag, very few students will ever put into practice the attacks they're being taught to defend against [6]. This emphasis on defense while largely ignoring the intricacies of attack may be most practical in the short-term view it hurts students in the long run. When Penn State Network Security students were asked the question: "Do you feel an understanding of techniques a malicious hacker would use will benefit you if you go to work in the Information Assurance field?" they unanimously agreed that understanding hacker techniques would further their ability to properly secure information. The same survey also brought to light that a majority of the students actually took personal time after the conclusion of the attack lab sessions to do their own investigation into the tools taught. With this in mind, combined with how little is actually taught about the intricacies of the attacks they eventually will be forced to defend against, students realize that their education omits an important issue of security. In short, defenders can better defend when they have a firm, technical foundation into how the attackers will attack.

4. JUSTIFICATION AND DUE DILIGENCE

With students themselves clamoring for experience in this area the question truly becomes is it necessary? While courses should stimulate students' interest these courses should also be applicable to the students after graduation. In my personal experience I have found skills in evaluating technologies and threats have been far more useful in the business world than my ability to configure appliances. Though defending the network is always the goal, training, firewall configuration, support for setting up VPNs, and instructions for getting an Intrusion Detection System tuned are available. When a company purchases such expensive technologies support for making them work is included. It takes a unique blend of experience, knowledge, contacts, and luck to be able to identify, analyze, and advise on today's threats, this is a combination that is difficult, if not impossible, to teach. [2] Students can however be given a framework from which to continue their learning, and for that it is necessary to teach lab exercises detailing the use of malicious tools. Understanding the configuration of a device is simple, understanding the potential impact of an exploit or tool to an enterprise network is complicated.

Undeniably teaching malicious tools is a double-edged sword. There is a fine line between teaching students to understand a possible attack and giving them the knowledge to carry out such an attack [4]. With every hacker tool that is taught educators risk creating the very menace they are trying to teach students to defend against. This paradox begs the question, how is it that these tools can be taught without simply giving the defenders more to defend against. In order to properly educate students in both the concepts behind a tool and correct ethical use of the tool it takes more than simply demonstrating the tool, but also showing students how it can be used appropriately. [6]

5. PROPOSED SITUATIONAL LAB CURRICULUM

As a proposed solution to this dilemma development was started on a new form of lab exercise to find a way to not simply teach the "how" of using a tool, but also the "when, where, and why". These labs use situations, mostly through involving valid auditing practices where similar tools would be used by corporate or government security teams to test their various defense in an effort to straighten their current position, not to compromise a foreign network [2]. This provides a way to help students understand that a given tool, while inappropriate, and probably illegal, to use without permission, can be a valuable asset when trying to keep malicious attackers at bay. This gives students a perspective to see that a tool is just a tool, and it is the user's implementation of that tool which decides if it is used to improve and protect a given network, or attack and wreak havoc against an unsuspecting target. These situational labs also provide a more real life experience than simply setting up a router and answering a few facts in a corporate style threat analysis report as was the case in former labs.

The beginning of the lab is a fictional situation, or story, into which the rest of the lab takes place [5]. This story details a time when a tool might be used in a valid security improvement context. It may be a password strength audit for a tool like L0pht Crack, or testing the practicality of a newly installed firewall for a tool like NMap, or possibly doing a network vulnerability audit for a tool like Nessus. Students are asked to place themselves in the shoes of an Information Assurance Engineer at a fictional company and use these tools to find out how they can further protect the company. After a brief set of open-ended instructions that encourage exploration and experimentation students are given a set of goals that force the students to properly utilize the tool in the laboratory network environment. These goals fit the context, and help guide the students in solving the problem posed in the context. Students might have to use L0pht Crack to conduct their password strength audit to find what employees are failing to comply with company password policy. They might need to identify unnecessary open ports that the firewall technician failed to lock down by scanning it using NMap, or they could be asked to identify what machines failed to receive the latest Microsoft Update and still have critical holes using Nessus. Each context and problem is tailored to the tool being taught, giving students a perspective on "when, where, and why" to use each tool.

Further reinforcement comes in the changes to the format and goals of the homework given at the end of each lab. Where past labs have had simple questions about typical commands or overarching concepts the new situational labs will continue in the fictional corporate context and require the students to submit their own audit reports. This requirement forces students to think critically as they would when employed by any real world corporation. The L0pht Crack lab would have homework that helps students craft an audit report detailing the amounts of passwords that complied and did not comply with standard company password policy. Furthermore students would create a list of users failing to comply and provide recommendations for remediation of these problems, as well as possible improvements to the policy as it stands. For the NMap lab a detailed report of the firewall box itself, what ports are left open, and any changes that should be made to the firewall policy would be expected. To complete the Nessus lab a report of patched and unpatched machines would be expected. This gives more of a real world, applicable end to the lab by getting students to complete the audit as would be expected, with a report to management detailing their procedures, findings, and remediation recommendations.

After students progress through five such labs they are faced with a final capstone project before lab curriculum progresses to set of defensive lab with the same emphasis on real world context. Continuing with the idea of context being essential to properly help students understand the use of these tools students are presented with an interesting set of problems in what is by far the most open ended lab. Student teams are put in a situation where they are using the tools they have learned to conduct an open penetration test, given a target of significance on the network and must use the skills they developed using these various tools to take advantage of the vulnerable system. This is not a hack, but an audit, and as a result a similar report to the other tasks is required. Teams are asked to detail the approach they took, the steps and vulnerabilities they used to find their way to the goal, and most importantly, a detailed report of remediation and suggestions for securing this network against their attack methodology, as well as any other possible attack methodologies and approaches.

The use of context and treating the lab like a real exercise are what gives students the “when, where, and why” that is lacking in traditional, procedure based labs. While a step-by-step walk through of a tool might give a student the barest amount of understanding about how to use a tool it is only the application of that tool that makes it valid. Further, in the corporate world, the use of a tool is only as valid as the results it generates, thus it is doubly important that reporting is an integral part of these lab exercises, reinforcing both the labs content, and giving students practice at producing these real world style reports.

6. CHALLENGES AND OPPORTUNITIES

In creating a set of labs that mimics the context of real life security audits, realism is perhaps the most difficult, yet most necessary, characteristic to establish. Students must believe in the possibility of truly being faced with this type of challenge when they enter the working environment for them to fully apply themselves. As labs are continuously created they must constantly evaluate the realism and validity of the situation in which they are used, since simply teaching the tool without a realistic use situation defeats the entire idea of helping students understand the ethical use of these tools. Reporting templates must also follow the labs and uphold the continuity by making the reporting requirements accurate.

Realism in the lab environment, especially for the penetration testing exercise, is essential, making the environment setup incredibly complicated and requiring careful planning. Care must be given to make sure that the objective can be accomplished using whatever group of tools is taught in a given semester. Furthermore, multiple attack methodologies should be valid for accomplishing the given goal. A true to life diversity of clients, servers, appliances, services, and operating systems should be represented. This is complicated by the need to make sure vulnerable versions of operating systems and services are in place in the environment, and that valid attacks and exploits are made available to the auditors, either by pre-configuring their testing host box, or by making the appropriate files available on a resource server or disk.

The detailed reporting provides a challenge over past lab formats as TA’s are forced to not simply check objective, short answer questions, but instead to read and evaluate longer, detailed audit reports, where answers may vary greatly, but still be correct. This limits the ability for a professor to simply provide a key to the TA, and instead requires the TA to be extremely familiar and use their own judgment to evaluate the validity of various answers. Lab creators must provide clear guidelines that attempt to remove as much ambiguity as possible from their guidelines of answers and minimize the amount of subjective judgment that those grading the exercises must use.

7. CONCLUSION

Information Assurance educators are faced with a difficult task in the years ahead. With a constantly growing need for Information Assurance professionals and a continuously changing realm of security it is a struggle to maintain both the correct body of knowledge and ethical attitude. Teaching malicious tools provides an essential part of developing a holistic approach and understanding to security. By giving students an understanding of attack methodologies and by allowing them to utilize the same tools malicious attackers use, students develop the holistic understanding of security that they desperately need. By teaching these tools in a context where students can use them ethically they understand not only "how" an attack succeeds, but "when, where, and why" that same tool can be used to keep the bad guys out. Essentially participating in a class that teaches the "when, where, and why" is the sheriff's badge of the new Wild West.

This work was supported by NSF DUE-0416827, and by DoD IA Institutional Capacity Building at Penn State Grant.

REFERENCES

1. "Wyatt Earp." Wikipedia. 2005. 9/21/2005 <http://en.wikipedia.org/wiki/Wyatt_Erp>.
2. Patricia Y. Logan and Allen Clarkson. "Teaching Students to Hack: Curriculum Issues in Information Security." SIGCSE '05 vol 1 2005. p. 157.
3. Laurie Werner. "Teaching Principled and Practical Information Security." Consortium for Computeing Sciences in Colleges vol 1 2004. p. 81.
4. Tom Wulf. "Teaching Ethics in Undergraduate Network Security Courses: The Cautionary Tale of Randal Schwartz." Consortium for Computing Sciences in Colleges vol 1 2003. p. 90.
5. Ed Crowley. "Information System Security Curricula Development." CITC4 vol 1 2003. p. 249.
6. James Harris. "Maintaining Ethical Standards for a Computer Security Curriculum." InfoSecCD Conference vol 1 2004. p. 46.

THE CYBERCIEGE INFORMATION ASSURANCE VIRTUAL LABORATORY

Mike Thompson

Naval Postgraduate School

Abstract: CyberCIEGE enhances information assurance education and training through the use of computer gaming techniques. In the CyberCIEGE virtual world, users spend virtual money to operate and defend their networks, and can watch the consequences of their choices, while under attack. These tutorial sessions will introduce CyberCIEGE to educators and information assurance training professionals. The IA training tool is provide to develop new scenarios.

Key words: Information Assurance, Education, Simulation, Virtual World

The CyberCIEGE project creates an Information Assurance (IA) teaching/learning laboratory. In addition to rigorous scientific foundations, it involves the application of abstract principles to the real world. A hands-on virtual laboratory provides a dynamic and often surprising context where abstract principles can be applied and discovered.

CyberCIEGE is an innovative computer-based tool to teach network security concepts. The tool enhances information assurance education and training through the use of computer gaming techniques such as those employed in SimCity™ and RollerCoaster Tycoon®. In the CyberCIEGE virtual world, users spend virtual money to operate and defend their networks, and can watch the consequences of their choices, while under attack.

In its interactive environment, CyberCIEGE covers the significant aspects of network management and defense. Users purchase and configure workstations, servers, operating systems, applications, and network devices. They make tradeoffs and prioritization decisions as they struggle to maintain the ideal balance between budget, productivity, and security. In its longer scenarios, users advance through a series of stages and must protect increasingly valuable corporate assets against escalating attacks.

The CyberCIEGE encyclopedia of security concepts contains a wealth of information assurance knowledge. Users can read the encyclopedia, or watch its instructional movies!

CyberCIEGE supports many educational venues, from basic workforce awareness training to university classes. It can help organizations meet DoD Directive 8570 obligations for IA training, annual IA awareness refreshers, and appropriate IA education. CyberCIEGE contains support for the creation of tools to record and assess student progress. Best of all, CyberCIEGE is extensible.

CyberCIEGE includes a language for describing its security scenarios. Using this language, educators may construct or modify scenarios that can then be played by students. CyberCIEGE was created by the Center for Information Systems Security Studies and Research (CISR) at NPS, and Rivermind, Inc., of San Mateo, CA.

1. WECS CYBERCIEGE INFORMATION ASSURANCE TRAINING TOOL TUTORIAL

These tutorial sessions will introduce CyberCIEGE to educators and information assurance training professionals. A hands-on laboratory will allow participants to explore the IA training tool and learn to develop their own scenarios. Participants will receive a copy of CyberCIEGE for their own use. Workshop topics include:

I. Introduction to CyberCIEGE
- Central concepts and abstractions
- Tool navigation and scenario demonstration
- Hands on play of a CyberCIEGE scenario

II. Strategies for deploying CyberCIEGE for training and education
- Simple scripted training scenarios
- Virtual setting immersion scenarios
- Assessing student progress using automated log analysis

III. Development and enhancement of CyberCIEGE scenarios
- Use of the Scenario Development Tool
- Tutorial to develop a simple new scenario
- Understanding CyberCIEGE virtual attackers motivations and means
- Reliance on game engine behavior vs. scripting scenarios with conditions and triggers
- Story telling to engage the player¹s emotions
- Packaging multiple scenarios into a single campaign

TEACHING ASSURANCE USING CHECKLISTS

Keynote

Matt Bishop

University of California, Davis

Abstract: Industry, the government, and many other entities have expressed concern about the way schools are teaching programming. The central theme is that students do not learn how to program "securely." Several proposals to improve the quality of programming involve checklists, or items that that students must demonstrate mastery of in order to achieve the appropriate goal (such as passing the class or graduating).

Underlying these proposals is the goal of teaching *assurance*—building programs and systems that meet specific requirements—and making it a part of everyday work, as opposed to a specialized discipline applied only when there is time, or when there are specific assurance-related requirements. This raises several issues, especially when one is considering "security assurance."

What is "security?" The standard answer, "security is whatever the security policy defines it to be," does not provide the guidance needed to develop a new student's intuition about how to develop programs. Instead, one needs to focus on a specific set of security requirements. No single set covers all situations, but certain elements arise in many requirements. For example, the requirement that a program handle error conditions appropriately means that a student must write programs in which error conditions are detected. Buffer overflows may not cause security breaches (especially if availability is not at issue), but they do cause errors to arise. Thus, a better pedagogic question than "how can we teach students to write more secure programs" is "what principles should we be teaching students to enable them to write more secure programs, and how can we teach them how to apply these principles to specific situations?"

This formulation recognizes that the term "secure programming" is imprecise because it means different things to different people, and in different environments. It also adds to our burden because we must teach students to evaluate situations and determine what "security" means *in the specific case*. This requires teaching principles.

History plays a part in this. Early papers, such as the Anderson Report, described threats to systems and created a framework for addressing them. These are important not only because current technology implements many of the ideas, but also because the framework itself teaches one how to think about assurance. Consider the reference monitor, a "guardian at the gate" that controls access to a resource. This means that *all* accesses of the resource must pass through the reference monitor. Spaghetti code, which allows multiple paths of access without a rigorous examination of *why* each path of access must be present, leads to poor coding and (potentially) security problems. The reference monitor must be checked, to ensure it works correctly; this leads to the requirement of smallness, so it can be validated. Complexity may be required, but unnecessary complexity is a weakness. Striving for simplicity, of style if not of content, is the mark of a good programmer. Finally, if an attacker can alter the reference monitor, or the data upon which it relies, that subverted control will allow unauthorized access to the resource. Hence the programmer must guard against such tampering, by (for example) checking error conditions to prevent attackers from exploiting them. From one framework comes several ideas central to protection.

Over time, technologies change, and methodologies adapt to new circumstances. But principles do not change. They provide guidance for developing new methodologies and technologies. Because of the rapid pace at which our field, computer science, is evolving, focusing on principles and teaching students how to analyze situations, and apply these principles, is critical.

Checklists have a part to play in this process. There are many different types of checklists. Such lists can provide guidance or specific items to consider, and may be used by a student or a grader (auditor).

- A checklist can simply list items to prompt students' memory, for example "check for possible errors and handle them." The usefulness of this type of checklist depends entirely on the student's ability to translate the items on the checklist into the particular domain in which the student is working. This list provides guidance, and the student is expected to look beyond the list as appropriate.

- A checklist can list specific items that students are required to satisfy. These checklists are usually specific, for example "check the return values of all functions to ensure the function worked properly." These checklists are useful for reminding the student about details, but risk the student performing the items on the checklist without considering their appropriateness, and not looking beyond the checklist for items that need to be considered.

- A checklist can be used to aid in the evaluation of a student's work, or of the result of that work. Again, this checklist is primarily a guide to the grader or auditor, who is expected to translate the generality of the items into specific criteria appropriate for the work.

- A checklist can list specific items that the student's work is to satisfy. Here, the checklist requires the assessor to determine if the item is met. If so, it is checked off; if not, the item is marked unsatisfied. The set of satisfied (or unsatisfied) items determines the score.

The best checklists are derived from principles, and their items develop logically through the derivation of principles, methodology, and application to a particular domain, guided by experience of practitioners. This allows one to justify the checklist rigorously and to see how the principles strengthen the practice; assurance at its best.

The type of education being sought, and the type of environment in which the checklist is used, determines the appropriateness of any particular checklist. Used properly, a checklist can enhance a student's education; used improperly, that same checklist can hinder the student's progress, as well as fail to achieve the goals of that student's education.

ACKNOWLEDGEMENT

This paper expands upon ideas discussed in a column co-authored with Deborah Frincke [2]. I am grateful to her for the discussions that helped me sharpen some of these ideas, and for her encouragement.

REFERENCES

1. J. Anderson, *Computer Security Technology Planning Study*, Technical Report ESD-TR-73-51, Electronic Systems Division, Hanscom Air Force Base, Hanscom, MA (1974).
2. M. Bishop and D. Frincke, “Teaching Secure Programming,” *IEEE Security and Privacy* **3**(5) pp. 54–56 (Sep. 2005).

EDUCATING SYSTEM TESTERS IN VULNERABILITY ANALYSIS

Laboratory Development and Deployment

Leonardo A. Martucci, Hans Hedbom, Stefan Lindskog and Simone Fischer-Hübner

Department of Computer Science, Karlstad University – Sweden

Abstract: This paper presents a vulnerability analysis course developed for system testers and the experiences gained from it. The aim of this course is to educate testers in the process of finding security weaknesses in products. It covers the four steps of a vulnerability analysis: reconnaissance, research and planning, mounting attacks, and assessment. The paper describes in detail ten different laboratory assignments conducted within the course. For each experiment, an overview and a description on how to run the assignment together with the expected knowledge obtained are presented. In addition, a course evaluation and lessons learned are also provided.

Key words: IT security education, vulnerability analysis, assurance, laboratory assignments.

1. INTRODUCTION

A constantly growing number of security vulnerabilities, threats and incidents has increased the efforts of IT and Telecom industry to invest in the development of more secure systems. A lack of security functionality and security assurance bears high risks for IT and Telecom vendors as it can, for instance, result in high costs for patching running platforms on the clients' sites, dissatisfied clients due to security breaches and system downtime, as well as reputation damage for the vendors' clients.

Vulnerability Analysis (VA) is an important means for improving security assurance of IT systems during test and integration phases. The approach that VA takes is to find weaknesses of the security of a system or parts of the system. These potential vulnerabilities are assessed through penetration testing to determine whether they could, in practice, be exploitable to compromise the security of the system. The Common Criteria have requirements on VA to be performed for the evaluation of systems with an Evaluation Assurance Level 2 (EAL2) or higher [0].

For improving security of their platforms before deployment, a major telecom company decided to increase their efforts in VA performed in their test labs, which requires well-educated software testing personnel. This company decided to outsource the education and training of its employees in the area of VA to an academic institution with experience in both VA and education. For that purpose, a compact VA course was developed at our department to be held during three working days for an international and heterogeneous, in terms of knowledge in the security area, group of software testers from industry. The emphasis of this course was put on practical, hands-on experiments in VA. The course was first held in spring 2005, with 16 participants, and two times more in fall 2005 with 15 and 14 participants, respectively.

Penetration testing had been employed by our group in the past to evaluate network-based computer systems. Our intent with the test performing was primarily to find quantitative measures of operational security. Students from an applied computer security course were engaged and trained to attack a target system and evaluate its security [0]. Experiences and lessons learned from these supervised student experiments provided us with some inputs for the preparation of this VA course.

This paper is organized as follows. First, the course content is briefly presented in section 2. Second, we provide a detailed description of the hands-on experiments, explaining the purpose and goal of each laboratory assignment and the knowledge that participants should obtain from those experiments in section 3. We present the lessons learned from the teachers' point of view and a summary of the course evaluation in section 4. Finally, we present our concluding remarks in section 5.

2. COURSE CONTENTS

As mentioned in section 1, the VA course described in this paper was developed primarily on request from the industry. The requested course was aimed for software testers with no or little knowledge about security in general and vulnerability analysis in particular, but with an extensive knowledge in software testing. Since the target group was practitioners, a practical course was requested.

The list of topics included in the VA course was, naturally, agreed with the contractor. The contractor had presented us a preliminary list of topics that should be covered by the course. This list was later transformed into a content list and a laboratory set that was discussed with a contractor's group composed of software testers, software designers, managers, industrial researchers and security experts. A three-day course (24 hours course) was selected as the best choice for the course length, since the testers should not be absent from their tasks for a long period. The course also includes a follow-up one-day recycling class after one year to update the participants about newly found vulnerabilities and VA tools. Approximately 30 to 40% of the course covers theoretical aspects and 60 to 70% is used for practical assignments. The latter was intended to give the attendees hands-on experience on how to conduct a VA of either a software component or a complete system. The course is structured in four blocks covering the theoretical part with hands-on assignments providing support to the theory. The course is divided into the four following blocks:

a) Introduction to Computer and Network Security. This block includes motivation, terminology, evaluation criteria, an overview of security standards, risk analysis, ethics and course rules.

b) Computer and Network Security Protocols and Tools. It covers cryptography, pseudo-random number generation and testing, public key infrastructure (PKI), firewalls, intrusion detection systems (IDS), virtual private networks (VPN), IPSec, SSH, and SSL/TLS. This block also includes the assignments presented in sections 3.3 to 3.5.

c) Vulnerability Analysis. An in-depth description of VA four distinct steps, i.e.: reconnaissance, research and planning, attack mounting and assessment.

d) Known Vulnerabilities, Reconnaissance Tools and Information Gathering. This block includes common host attacks, a short section on viruses, worms and anti-viruses, system hardening strategies and several practical assignments presented in sections 3.6 to 3.10.

The last assignment is a full-day final project final practical assignment that concludes the course, a “putting it all together” experiment that summarizes the full vulnerability analysis process.

3. HANDS-ON ASSIGNMENTS

In this section, we justify the selection of this list of experiments, present the learning goals and also describe how the laboratories were deployed, i.e. network topologies used, applications needed and operational systems employed. In addition, we describe how the knowledge obtained can be applied in the students’ working environment in order to improve the security of a delivered product. However, we first introduce the ethical rules, the laboratory environment and the preparations that had to be done before running the course. The laboratories are described in this paper in the same order as they were presented during the course, in order to follow the theory presented in parallel.

3.1 Ethical Rules

VA course teachers clearly instructed the participants that VA strategies and penetration tools must be used for testing purposes and controlled environments only, in order to avoid intentional or unintentional harm to others. Hence, for this VA course we set up the following ethical rules that participants were requested to adhere to:

– Do not experiment with VA-tools without explicit permission of an authorized party.

– Do not pass on/publish material, tools, and vulnerabilities to unauthorized parties.

– Do not use your technical skills in criminal or ethically questionable activities.

– Always report flaws to vendors/developers first.

– Software tools provided in this course must only be used in a laboratory environment and on laboratory computers.

3.2 The Laboratory Environment

The laboratory is prepared for up to 20 students working in pairs. The workstations are dual boot - Windows XP/Linux and Fedora Core 3 Linux distribution - Intel Pentium 3 900MHz with 256MB of RAM, with one auto-sense Ethernet/Fast Ethernet network interface card (NIC). Two additional workstations were configured as servers – one running Windows Server 2000 and one running Fedora Core 3 Linux – running various services. Servers are used in some, but not all, laboratories as target systems. Moreover, the server machines are also used to store image files for the other workstations, in order to fast redeploy images if necessary.

3.3 Password Cracking

Passwords are a very basic and common authentication method. Unfortunately, passwords are not the most secure way to implement authentication, but, on the other hand, passwords are the most practical and also less expensive way to implement it. Password policies are out of scope of this assignment, even though they should be mentioned during the theoretical part of a VA course. During the implementation phase, it is common that developers use some default passwords for debugging and testing their applications before being delivered to integration and test teams in order to speedup the development phase. However, it is not uncommon that those debugging and test passwords are left in the application after the development phase is over. Those development and debugging backdoors are a serious threat to deployed platforms, as they can even grant full privileges. Detecting those built-in passwords is a must activity for testers. In this experiment, we go one step beyond detection, as students are encouraged to crack password files, in order to evaluate how easy it can be achieved with open-source tools.

3.3.1 Running the Assignment

In this experiment, students run a local password cracker application. There are several open source tools available for this purpose – our choice was John the Ripper v.1.6, a password cracker tool which main purpose is to detect weak UNIX passwords [0], since it is an easy to use and easy to deploy. We also provide synthetic (artificially populated) *passwd* and a *shadow* files from a Linux box. Before starting to use the tool, we provide students with some well-known rules that are used to choose good passwords, such as substituting letters for numbers, how to identify if passwords are encrypted with *crypt*( ) or hashed with MD5 and also the structure of password files. Finally, the students just have to *unshadow* the *passwd* file and run the John the Ripper tool. Since running the password cracking tool can take long time to run, it is advisable to have easy passwords added in the password file. Therefore, students can extract some passwords from the password file quickly.

3.3.2 Knowledge Obtained

Even though the experiment is more focused on cracking passwords, developers should first be able to detect if there are backdoors in the system, and, if possible, correct the problem and document it.

3.4 Testing for Randomness

Pseudo-random number generators (PRNG) are used in several applications with a very broad scope, from simulation and numerical analysis to gaming applications. PRNG are also a crucial cryptographic primitive and a flawed PRNG implementation can compromise the security of a whole system. A classic example was a PRNG flaw in earlier versions of the Netscape browser that could compromise the security of SSL connections [0]. Well-known PRNG have, in most of the cases, public algorithms and are submitted to several batteries of tests. In addition, the U.S. National Institute of Standards and Technology (NIST) also has a list of approved random number generators applicable to FIPS PUB 140-2 standard [0]. However, it is possible that a programmer might use a flawed implementation of a given PRNG. Therefore, it is crucial for testers to identify weak binary sequences produced by a PRNG. This is the main goal of this assignment.

3.4.1 Running the Assignment

In this experiment, the NIST Statistical Test Suite [0] is used to evaluate outputs from different PRNG. It runs in Windows 32bit systems. It implements 15 statistical tests and also has embedded implementations of well-known bad PRNG, such as the XOR RNG, and also NIST approved RNG, such as the ANSI X9.31. Sample data files are also provided as companion part of this test suite. Even though other test suites, such as DIEHARD [0] and ENT [0] could be used, the NIST test suite is the latest and most complete test for randomness. Moreover, installing and running the NIST Statistical Test Suite installation is straightforward and it is also freely available.

Since generating sufficiently large files of random numbers is a time-consuming activity, the sample files available with the NIST Test Suite are used, since outputs from good and also bad PRNG are provided. However, our recommendation is to produce files generated using Java classes *java.util.Random* and *java.security.SecureRandom* beforehand and also with the *rand* function from C. Therefore, students can evaluate the output quality of those PRNG, which are popular among developers.

Finally, since analyzing the outputs of the NIST Test Suite demands basic knowledge of statistics, it is crucial to provide to the students a short background on hypothesis testing before running the test suite. It is also important to briefly explain the tests and the test requirements as well, since tests have different requirements regarding the minimum number of samples and the minimum length (in bits) of each sample.

3.4.2 Knowledge Obtained

In order to evaluate if a given PRNG provide weak binary sequences, developers should inform and provide methods to call random number generator objects or functions in order to test them (if any are used) or, at least, document that a given PRNG is being used for security reasons. In addition, it is possible to verify if a given implementation of a given PRNG class produces weak binary sequences or not, and, eventually, validate a PRNG class. Furthermore, testing for randomness can be automated.

3.5 Firewall

Firewalls can be briefly described as filters that follow some screening policies defined by the network administrator. There are several firewall applications and boxes available in the market and even open source firewalls, such as *ipTables*. It is of ultimate importance for system integration teams to know how firewalls work, where they should be placed in a network topology and how their rules are defined and verified. The goal of assignment is to provide hands-on experience on such aspects.

3.5.1 Running the Assignment

This exercise is based on an assignment originally published in [0]. Students are divided into groups of two. Each group is given a description of a setup as the one described in Figure 1 and are asked to write firewall rules in Linux using *ipTables* implementing a given policy defined in the problem statement. Rules are written as if all the necessary NIC were present. Note, however, that the students are only working with virtual interfaces and not with two separate NIC.

3.5.2 Knowledge Obtained

When deploying systems that have a firewall, testers should be able not only to test the system from a black box point of view, but also read, understand, verify and evaluate firewall rules, in order to look for inconsistencies in the access control list. Therefore, hands-on experience is fundamental for testers in order to perform such tasks.

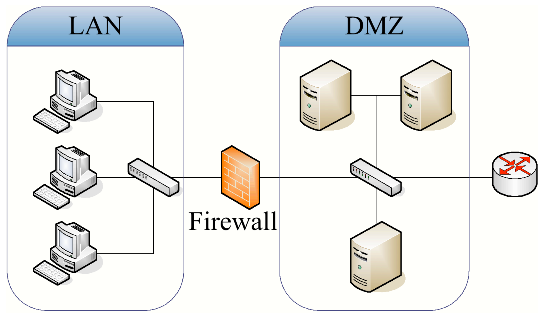


Figure 1. Network topology for the firewall assignment.

3.6 Black Box Test Method

Communication protocols are basically defined in open and proprietary standards. Those standards are usually well-known and were evaluated by a committee and also by the users' community. Therefore, finding out shortcomings in standards is not an everyday event. However, finding out vulnerabilities in implementations of communication standard is not uncommon since standards provide guidelines for implementing, but not for the coding. Thus, careless programming and also coding errors can lead to a series of vulnerabilities that can compromise a running system. Testers should therefore be able to detect such flaws in the communication protocols during test and integration phases. There are basically two ways to test the implementation of a communication protocol. The first is to extensively review the source code in order to find out implementation flaws, but this method is cost and time inefficient. The other alternative is the black box testing approach, i.e. a functional test method that evaluates a system from the input and output point of view in order to identify system vulnerabilities. It is important to testers not only to be familiar with those tools but also how to interpret results and draw conclusions from them. In this experiment, the students are encouraged to work in larger groups and execute a black box testing against a running system.

3.6.1 Running the Assignment

In this experiment the PROTOS tool [0] is used as a black box testing tool against the SNMP protocol implementation of Cisco 1005 router running IOS 11.1(3). The PROTOS Test Suite c06-snmpv1 is used here to perform a Denial of Service (DoS) attack against the Cisco router. PROTOS c06-snmpv1 test suite [0] is divided into four separate test material packages: two test packages regarding packets with encoding errors and other two test packages regarding packets with application exceptions. Since PROTOS is a Java-based application, any OS can be selected, as long as a Java Virtual Machine is installed. Students were divided in two groups - each group with four workstations. The network topology used in this assignment (for each group) is shown below in Figure 2.

The goal of this experiment is to check if the SNMP implementation of this router is vulnerable to a malformed packet generated using the PROTOS test suite and also identify the vulnerability type (i.e. test material, test group and test case), if it exists. PROTOS c06-snmpv1 test suite documentation [0] is used to identify test groups. It is useful to provide a short introduction of the SNMP protocol PDU in order to provide some background information on this protocol.

In this assignment students are encouraged to cooperate in order to find out if the router is vulnerable or not to SNMP malformed packets. All workstations continuously ping the router in order to check if it alive or not. No console access to the router is provided until the test case is identified. After the test case is identified – in the particular case of this router running the IOS

11.1(3), it is a basic encoding rule (BER) error in the length field of a Get-Request SNMP PDU – students are allowed to connect to the Router using the console port and witness the router crashing after receiving just one malformed packet. Basically, PROTOS just send one test case after another, but the running process doesn't stop if the target system misbehaves. Therefore, students have to keep track of the echo reply messages, and follow PROTOS execution to finish this assignment.

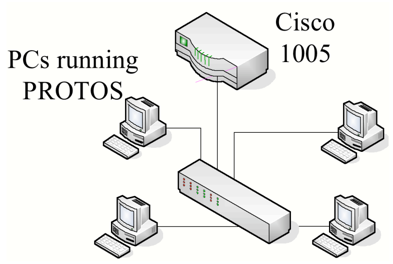


Figure 2. Network topology for the vulnerability test using the black box test method assignment.

3.6.2 Knowledge Obtained

Black box test strategy is very successful method to identify vulnerabilities in implementations of communication protocols. Automated testing tools are especially useful in the security evaluation of a given system because an entire test battery can be executed with a single command. The PROTOS test suite is a freely available tool, but few communication protocols are available – WAP, HTTP Reply, LDAPv3, SNMP, SIP and H.225.0. Additional test suites are available within Codenomicom [0], the commercial version of PROTOS. Using such tools, testers can quickly evaluate the robustness of protocol implementations and report vulnerabilities found back to the development team in order to fix them.

3.7 Network Analyzers and ARP Spoofing

Network analyzers are tools used to capture and analyze network traffic being transmitted in a given collision domain. These tools simply set the network interface card to promiscuous mode in order to bypass the MAC filtering rules. They are also very common tools for testers to evaluate the network traffic received and transmitted by a given system. However, testers also need to understand that some weaknesses are inherent to the protocol design. It is well-known that some protocols were designed with very little concern about security, such as FTP, TELNET and etc. Although the decision of using such protocols is not up to test and integration teams, they should be able to identify and report the existence of such protocols. A second part of this experiment is prepared in order to prove to students that knowledge of the networking device internals and network topology can impact security analysis. In the same assignment we also present an ARP spoofing experiment. The reason for this experiment is to prove to the students that some general affirmations about security are not true, and that skepticism and hands-on knowledge (in this case networking) are very important points in the security area. In this experiment we prove that the following assertion is incorrect: "Switched-based networks are obviously protected against network analyzing because each switch port is one collision domain". Although this assignment is basically related to network security, its main goal is to prove that knowledge in network environment (i.e. topology and protocols) is definitively important to assess system vulnerabilities.

3.7.1 Running the Assignment

The assignment is divided in three parts, each part with a different network topology. In the first part, students verify that a very popular network protocol, the TELNET, is very weak regarding security using a network analyzer tool. The main goal of this part is to introduce a network analyzer tool to the students. In the second part, a rather insignificant change in the network topology modify test results – and it is up to the students to figure out why changes occurred – and, finally, in the last part of this assignment students test an ARP spoofing tool on a switched network, verify the achieved results and explain those results according. In this assignment we deployed in all workstations Ethereal v.0.10.11 [0], an open source (license under the GNU GPL) network analyzer. Ethereal source is available for download, and installers are available for Linux Fedora, Unix Solaris and Windows 32bits systems.

Ettercap [0] and Cain & Abel [0] are two flavors of ARP spoofing tools. Ettercap is an open source tool (source code is available, but no binaries) and have more features than Cain & Abel. On the other hand, Cain & Abel runs only in Windows 32bits platforms, but has a nicer GUI and it is easy to install, in addition, Cain & Abel is a freeware, but no source code is available. Therefore, the OS choice is up to the instructor. We considered that a better GUI is more important than a more powerful tool, at least from the didactic point of view, and, therefore, we picked Ethereal, Cain & Abel in a Windows 32bits environment. The three different scenarios created for this assignment are detailed next.

3.7.1.1 1st Part: Network Analysis I

The first part of the assignment has two goals: introduce the network analyzer (Ethereal) through a very simple scenario and also demonstrate that the design of some protocols have not taken security as a requirement – TELNET, a popular terminal emulation application is used to prove the assertion. All workstations and a Cisco 1005 router are connected to a 10Mbps hub. The instructor task is to access the router via TELNET. The network topology is depicted below in *Figure 3*.

The experiment goal is to use the network analyzer to capture the access password and the privileged user password. Even though this is a very simple exercise, instructors should teach beforehand how to set and write filter rules in Ethereal. In order to demonstrate the importance of filtering, the instructor should keep a regular background network traffic – that can be achieved with a continuous ping from the instructor workstation to the router, for instance.

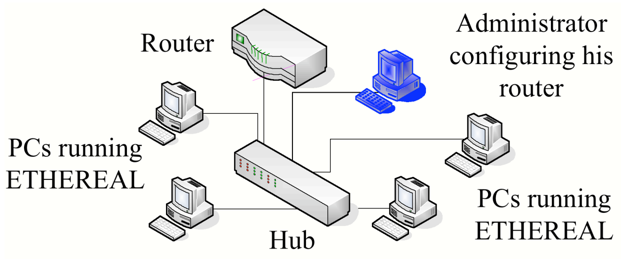


Figure 3. Network topology for the network analysis assignment - Part I and Part II.

3.7.1.2 2nd Part: Network Analysis II

The goal of the second part of this assignment is to demonstrate that knowledge of network device and its internals can make difference on a VA test. The scenario is basically the same of the first part of this assignment, presented in *Figure 3*: a system administrator is configuring a router using TELNET. However, the network topology has slightly changed: the 10Mbps hub was upgraded by the system administrator to a 10/100Mbps auto-sense hub.

Before starting to access the router via TELNET, the instructor must force his network interface card to 10Mbps (without students noticing it). Therefore, TELNET traffic from the instructor workstation to the router will be send in the 10Mbps backplane. Since the NIC of the student workstations will be automatically set to 100Mbps, no TELNET traffic will be transported in the Fast Ethernet backplane of the hub, and, therefore, they will not be able to capture the traffic exchanged between the instructor's workstation and the router, but they will be able to verify that the router is up and running using ping or TELNET. Students are then invited to think and explain why such subtle change in the network topology had dramatically changed the achieved results. The answer lays in the internals of the auto-sense 10/100Mbps hub.

Since the auto-sense 10/100Mbps hub has two independent backplanes, one running at 10MHz and the other at 100MHz, it also has two independent collision domains connected by a layer 2 bridge. Thus, the traffic between the router and the administrator's workstation is transmitted only on the 10MHz backplane, but this traffic is not forward to the 100MHz backplane, where all student workstations are connected.

3.7.1.3 3rd Part: ARP Spoofing

The goal of the third part of this assignment is to prove that some common assertions on security can be proven wrong with a clear understanding of the subject – ARP (Address Resolution Protocol) protocol in the case of this assignment. A simple star topology is used, with student workstations interconnected to a switch. Since one MAC address shall be spoofed by only one machine per time period in order to capture the all traffic flow, and in order to allow all students to work in parallel, student workstations will spoof other student machines, in a logical ring topology. In other to create a constant traffic flow, students were instructed to continuously ping the workstation to their right or to their front.

The target assigned for the ARP spoofing experiment is the ICMP echo request traffic flowing from the next workstation to the right to the one next to it. Therefore, students send to the next workstation to the right ARP reply PDU in order to update its ARP table with the attacker's MAC address instead of the MAC address of the recipient. *Figure 4* presents the ICMP echo-request PDU traffic flow chain, the ICMP echo-request target flow, the attacker and target workstation.

The logical chain topology depicted in *Figure 4* allows all students to work in parallel in order to optimize the time to run the assignment. With such kind of topology, students play the role of attackers and victims at the same time, since while they are poisoning the ARP table of their neighbor to the right, the other neighbor is poisoning their ARP table. Cain & Abel was used as ARP poisoning tool in parallel with Ethereal, which was used to capture traffic in the network segment. However, the main goal of the experiment is not just the understanding of how such attack is performed, since it is basically straightforward as soon as the students were told that ARP is a stateless protocol.

3.7.2 Knowledge Obtained

Network analyzers are powerful tools that testers should be able to master, since they can provide a good insight of the data being transferred from and to a running system. However, only mastering such tools is not enough without knowledge about the test environment, as proved in the second past of this assignment, and about the protocols involved, as demonstrated in the ARP spoofing experiment. In addition, ARP spoofing is an easy to detect attack. The goal of this assignment is to demonstrate that a lot of myths in the security area are not true, and those myths can only be proven wrong with a good understanding of the target system and its running protocols. In conclusion, it is fundamental for testers to have a deep understanding and knowledge of the test environment and of the running system especially when testing for security.

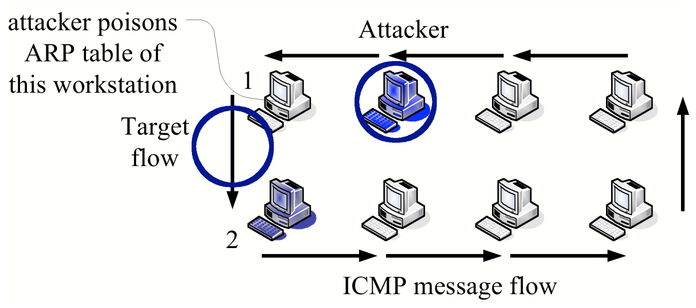


Figure 4. The ICMP echo-request traffic flow.

3.8 Port Scanning

Port scanners are used in the network reconnaissance phase of VA. Such tools basically scan one or more devices in order to find open ports that could be exploited for an attack. It is a fundamental task for testers to evaluate if unexpected open ports can be found in a system, especially during the integration phase of a system, just before deployment.

3.8.1 Running the Assignment

The goal of this experiment is gathering information about two running systems. Two workstations configured as servers (one Linux and one Windows server) are used as target systems. Students run Network Mapper (NMAP) [0], an open source port scanning tool, under Linux (however, it can be also compiled for Windows 32bit, since the source code is available). Servers run several network services, such as FTP, HTTP, NetBIOS (Windows server), etc. Students have to find and identify the servers in a given IP network range, since their IP addresses are not provided, find out the operational system running and identify the open ports in each server. The network topology employed is a simple star topology with all workstations and servers belonging to the same broadcast domain and in the same IP subnet.

3.8.2 Knowledge Obtained

Port scanners are reconnaissance tools usually employed by attackers in order to gather information about a system and identify open ports that can lead to successful break-ins. Testers should use such tools is order to identify unexpected open ports (i.e. ports that are open, but should not be), report them, and, eventually, terminate a process in order to close those ports during system integration phase.

3.9 Node Hardening

Part of the testing process is to test systems that are to be delivered and to do system tests of product. In that phase it is desirable that the testers make sure that the products are as secure as possible. One step in this securing process is to be sure that no unnecessary services are running and that the services needed run with minimal rights only. It is also vital to be sure that all the software used is patched to the latest patch level. The whole process is known as host hardening. Host hardening can be conducted in several ways, such as using checklists, experts in the area or software tools.

3.9.1 Running the Assignment

Since the students have a heterogeneous background regarding in-depth knowledge of different operating systems, an open source tool for the hardening process was selected for this assignment. The chosen tool was Bastille [0]. It is a node hardening software that lets the user answer a number of questions on how they want to be configured and then configures the system according to the answers. In essence it works as a high level automated checklist. It can also be used to analyze the current configuration and give suggestions on detected suspicious or dangerous configurations. Students were asked to make there own computers as secure as possible given that it still should be functional as a networked computer. They were also told that, later on, they will use a security scanner to test how secure their systems were (section 3.10). The goal of this exercise is to provide to the students an understanding of the host hardening process and the phases that are part of this process.

3.9.2 Knowledge Obtained

Even though there are tools that can be used in the hardening process they still rely on in-depth knowledge of the system that is to be hardened. It is also essential that these tools are up to date with the latest configuration files. Even though they can be a great help in the hardening process and also in the later verification of the hardening there are still a large manual portion of both these steps that require practical system knowledge. A system tester needs to know that in order not to be overly confident in the reports that these tools generate.

3.10 Security Scanning

A security scanner is a tool that attacks, or simulates attacks, on a system. The attacks it will perform are kept in a configuration file. The more up-to-date this configuration file is, the more accurate the scan. This means that an up-to-date security scanner will scan the system for all known vulnerabilities from the outside. Some scanners also have components that make it possible to perform attacks both from the inside and from the outside. A number of flavors exits but the most known ones are IS [0], Retina [0] and Nessus [0]. These tools are often a good starting point when doing a VA of a system since they automate the testing of the known weaknesses. The problem with these tools are generally not running them but rather how to configure the attacks and how to interpret the results, since false negatives and false positives in the attack process is highly influenced on the setup of the tool.

3.10.1 Running the Assignment

In this exercise, two target servers were set up. One Windows 2000 (set up as a domain server) and one running Fedora Core 3 Linux. Both not patched, running all standard services and acting as target systems for the security scanner. The Windows server ran IS and the Linux server ran Nessus. Neither the configuration nor the IP addresses of the servers were given to the students. The students were divided into groups of two. All the groups were asked to find the servers IP addresses and to find out which operating systems they were running. Students were also told that the servers were the only two network devices running an HTTP server in the network. After finding the servers, the students had to scan them and report all the vulnerabilities found. Finally, they were also requested to scan their own workstations in order to verify the hardening process that was performed earlier (section 3.8).

3.10.2 Knowledge Obtained

These tools are a good help to the testers in the verification process. However, they cannot be fully trusted and will inevitably generate both false positives and false negatives. A good knowledge of the system is needed in order to be able to correctly interpret the tool outputs. A tester also needs a good understanding of the tool in order to configure it correctly and customize its behavior if it is to be used on a non-standard system or with non-standard services.

3.11 Putting it all Together

Finally, we let the students put all their knowledge together in a final “grain to bread” exercise. Our goal was to make this assignment as close to their reality as we could, in a controlled environment, with limited resources and time.

3.11.1 Running the Assignment

In this exercise, the students were divided into groups of four. Each group was given a Fedora Core 3 Linux server with all services running. Every group also got a requirement document describing the role of the server and the security requirements on it. They were asked to find out what needed to be done in order to fulfill the requirements and also to perform the changes and verify the results. In order to do this they had access to all the tools used in the previous exercises and a list of useful internet links. They also had access to the Internet and were told that they could use it freely. Moreover, they could also use any other freely available tool found on Internet. The students had to report all the miss-configured parameters, vulnerabilities and also had to suggest changes in the system in order to adhere to the specification. The results were discussed in a summing-up session just after the exercise.

3.11.2 Knowledge Obtained

This assignment provided to the students a wider insight of the highly creative art of VA and it further reinforced their conclusions from previous assignments that the combination of good system knowledge, good understanding of the tools internals and their shortcomings are essential in order to correctly interpret and test a system.

4. EVALUATION AND LESSONS LEARNED

In this section, we present the course evaluation according to our feedback from the students. We also present the lessons learned from the teachers and university staff’s point of view.

Just after the course conclusion, the students received a questionnaire used to evaluate the course and the usefulness of the laboratories. The students were invited to individually answer the questions following a scale from 1 (lowest grade) to 7 (highest grade). They were also invited to comment their choice. No identification field was included in the questionnaire in order to provide anonymity to the students. A summary of the achieved results is provided in *Table 1*, regarding 36 returned questionnaires.

Table 1: Questionnaire summary.

Questions:	Average Grade	Standard Deviation
Evaluation and usefulness of the assignment:		
Security scanning	5.8	0.9
Port scanning	5.8	0.9
Node hardening	5.6	1.1
Black box test method	5.4	1.1
Password cracking	5.4	0.7
Network analysis (Parts I and II)	5.4	0.9
ARP spoofing	5.2	1.2
Putting it all together	4.8	1.2
Firewall	4.4	1.2
Testing for randomness	4.0	1.7
Did the course fulfill your expectation?	5.1	1,1

According to Table 1, it is possible to verify that seven out of the ten assignments of the course were highly classified by the students, being graded in the interval 5.2 and 5.8. They were: security scanning, port scanning, node hardening, vulnerability analysis using the black-box method, password cracking, network analysis and ARP spoofing. The reasons reported by the students included the usefulness of these assignments in real test environments and the easy and fast deployment in the tests. However, some participants were already familiar with some security tools and others reported that some of these assignments were not particularly useful in their specific tasks.

Firewall, testing for randomness and the final assignment were considered the least interesting assignments, being graded between 4.0 and 4.8. Some considered that the time reserved for the final experiment was short, while others thought that it was no more than a sum-up of the previous exercises. However, part of the students considered it a good concluding exercise. The firewall assignment was considered not that useful because testers hardly evaluate or write access-list rules in general, so this experiment was down rated by the participants compared to other assignments. Finally, testing for randomness was considered hard to be implemented and most of the students were not comfortable with their background in statistics. Indeed, testing for randomness is a test were a security primitive is being tested, which is usually underrated by testers, that are more concerned in securing systems in an upper abstraction layer.

To the question “did the course fulfill your expectations?” the average obtained grade was 5.1 (with a standard deviation of 1.1), what was considered very good result from the teachers and the contractor’s point of view, regarding the heterogeneity of the audience.

From the course backstage side, we noticed that having a system administrator available is essential during the course in order to provide assistance for the laboratories setup. This may alleviate the burden over the teachers, since a system administrator can redeploy images in the workstations or rebuild the network topology while teachers are explaining the assignments.

5. CONCLUDING REMARKS

This paper presented the outline of a practical vulnerability analysis course developed for software testers with emphasis on the descriptions of its laboratory assignments. The course evaluation clearly shows that participants were very satisfied in general with the course, and we are convinced that the course has significantly raised their awareness concerning network and computer security in general and of vulnerability analysis in particular.

However, we are not yet sure to which extent they use their new knowledge in their daily work with software and system testing. In spring 2006, during the one-day follow-up recycling class, we intend to investigate how the participants from the first course used their knowledge in vulnerability analysis when testing software products. In conclusion, we expect that the guidelines presented in this paper and the laboratory assignments are a very good reference for other academic institutions and industry to start a vulnerability analysis course in their premises in order to increase security awareness of their students and personnel.

REFERENCES

1. Common Criteria. Common Criteria for Information Technology Security Evaluation. Version 2.2. Jan. 2004. Available at: <www.commoncriteriaportal.org/>. Accessed in: Aug. 29th, 2005.
2. Lindskog S., Lindqvist, U. and Jonsson E. IT Security Research and Education in Synergy. In: L. Yngström, S. Fischer-Hübner (Eds). Proceedings of the IFIP TC11 WG11.8 First World Conference on Information Security Education - WISE'1, p.145-162. Kista, Sweden, Jun. 1999.
3. Openwall Project. John the Ripper Password Cracker v.1.6. Available at: <www.openwall. com/john/>. Accessed in: Jul. 25th, 2005.
4. Goldberg, I. and Wagner D. Randomness and the Netscape Browser. Dr. Dobb's Journal, n.260, Jan.1996.
5. National Institute of Standards and Technology. Annex C: Approved Random Number Generators for FIPS PUB 140-2, Security Requirements for Cryptographic Modules. Federal Information Processing Standards Publication 186-2. Draft. Gaithersburg, MD, USA: NIST. Jan. 2000.
6. National Institute of Standards and Technology. NIST Statistical Test Suite version 1.8. Available at: <http:// csrc.nist.gov/rng/rng2.html>. USA. Accessed in: Jul. 21st, 2005.
7. Marsaglia, G. DIEHARD Battery Test for Randomness. Available at: <www.stat.fsu.edu /pub/diehard/>. USA. Accessed in: Jul. 21st, 2005.
8. Fourmilab Switzerland. ENT – Pseudo-Random Number Sequence Test Program. Available at: <www. fourmilab.to/random/>. Switzerland. Accessed in: Jul. 21st, 2005.
9. University of Oulu. PROTOS Security Testing of Protocols Implementations. Available at: <www.ee.oulu.fi/ research/ouspg/protos/index.html>. Finland. Accessed in: Jul. 25th, 2005.
10. University of Oulu. PROTOS Test Suite: c06-snmpv1. Available at: <www.ee.oulu.fi/research/ouspg/protos/ testing/c06/snmpv1/>. Finland. Accessed in: Jul. 26th, 2005.
11. Codenomicon Ltd. Codenomicon. Available at: <www.codenomicon.com>. Finland. Accessed in: Jul. 26th, 2005.
12. Mixter. Gut Behütet. C'T - Magazin für Computer Technik. v.4, p.202-207. Heise, Germany, 2002.
13. Ethereal. Ethereal: A Network Protocol Analyzer v.0.10.11. Available at: <www.ethereal. com>. Accessed in: Jul. 27th, 2005.
14. Ettercap. Ettercap NG-0.7.3. Available at: <http://ettercap.sourceforge.net/>. Accessed in: Jul. 27th, 2005.
15. Oxid.it. Cain & Abel v.2.7.3. Available at: <www.oxid.it/>. Italy. Accessed in: Jul. 27th, 2005.
16. Insecure.org. NMAP - Free Security Scanner for Network Exploration & Security Audits. Available at: <www.insecure.org/nmap/>. Accessed in: Jul. 28th, 2005.
17. Bastille-Linux. The Bastille Hardening Program. Available at: <www.bastille-linux.org/>. Accessed in: Aug 31st, 2005.
18. Internet Security Systems. Internet Scanner. Available at: <www.iss.net/products_services/enterprise_ protection/vulnerability_assessment/scanner_internet.php>. Accessed in: Aug. 31st, 2005.

19. eEye Digital Security, Retina Network Security Scanner. Available at: <www.eeye.com/html/Products/Retina/index.html>. Accessed in: Aug. 31st, 2005.
20. Nessus Open Source Vulnerability Scanner Project. Nessus. Available at: <www.nessus.org/>. Accessed in: Aug. 31st, 2005.

COMPUTER SECURITY EDUCATION

Past, Present and Future

Carol Taylor, Rose Shumba, and James Walden

University of Idaho, Indiana University of Pennsylvania, and Northern Kentucky University

Abstract: This paper presents an overview of computer security education in academia. We examine security education from its recent past in the late '90's, evaluate its present state and discuss its future potential. A brief history of government programs that affect security education is presented along with their role as funding sources. The software industry perspective on security education and their relationship with academic security programs is also discussed. We define goals and objectives for the future of academic computer security programs and address barriers to successful achievement of those goals.

Keywords: Computer security education, computer science education, curriculum development, computer security

1. INTRODUCTION

Computer security became a tangible Computer Science sub discipline in the 1970's as the need to protect information became important with growing computer use in government and industry. At that time computer security research was funded by the military and primarily aimed at the protection of sensitive information. Computer security researchers and practitioners were few in number, worked primarily in the defense industry, and were mostly self taught.

Today, 30 years later, computer security is well established as an area of research and study within Computer Science. There are defined career paths for computer security professionals and an array of professional training and academic degree programs. If we compare the activity and interest in the field of computer security with its inception, one can say that a great deal of progress has been made. Yet, there is ample evidence that much more remains to be done.

Popular press reports describe daily the number of vulnerabilities found and the latest abuse of our systems by individuals in search of easy profit. Tumbleweed Communications, an e-mail security provider estimates that two-thirds of all e-mail is illegitimate traffic [1]. Botnets which consist of hacker-controlled networks of thousands of hosts are one of the fastest growing menaces on the Internet. These networks are capable of launching DDoS attacks, untraceable spam relays and widespread malware attacks [2]. SEI/CERT has stopped reporting incidents since they feel that widespread use of automated attack tools are so common that incident counts no longer provide meaningful information [3].

Several well-respected Computer Security researchers and educators also question the state of our knowledge and practice as a discipline. Roger Schell describes how the lack of science in computer security has actually led to a decline in the number of secure systems from a peak in the

1990's [4]. Eugene Spafford, also questioned the quality of security practice in his paper, *A Failure to Learn From the Past* [5]. Spafford recounts the 1988 Internet worm incident and points out that the same conditions that allowed the worm to wreck havoc on systems still exist in 2003 nearly 15 years later.

As security researchers, it is discouraging to see the low level of practice in the *real world* with the constant stream of new system vulnerabilities and the increasing number of malicious programs in search of one of the many unpatched systems. But, as educators, we are hopeful that in time, through education, we can improve the current state of computer security by producing students trained in secure coding, with knowledge of secure system design and operation.

While there are multiple studies in the security education literature that document experiences of individual departments in developing academic security programs, there is at present no general study of security education[1]. Our motivation for this paper is our belief that a current overview of computer security education is needed in order to assess overall progress within the discipline and offer possible future directions.

In this paper, we examine the state of computer security education from the past, present and future. We include views from three separate groups that have a strong interest in security education: academia, government and industry. We review the state of academic security education since 1997, the year of the first CISSE conference [6], in order to assess progress over the past eight years. We then evaluate the current state of computer security education plus provide personal insights from our respective programs in computer security. In the last part of the paper we discuss the future of security education in terms of goals and objectives and note possible barriers to progress.

The paper is organized as follows: this section provides background and our motivation for the paper, Section two examines the progress made in computer security education in the past eight years, Section three presents the current state of security education and research funding, Section four describes our respective experiences in security curriculum development, Section five discusses the future of academic computer security programs and Section six concludes the paper by highlighting the future status of computer security education.

2. COMPUTER SECURITY EDUCATION IN THE PAST

In this section we trace the evolution of government initiatives for academic computer security education over the past eight years. We provide statistics on the programs and the events in government that have influenced the overall progress of the early academic security programs. We also briefly mention the state of early academic programs.

2.1 Academic Programs

Early academic programs in INFOSEC education were primarily for graduate students. Consequently, undergraduates wanting to learn about computer security had to take graduate courses or do so through independent study. Graduate level security courses typically concentrated on Multi-level Security [2](MLS) concepts or simply covered cryptography without a lot of practical system analysis [23].

[1] There are two CISSE keynote speeches by Matt Bishop in 1997 [23] and 2000 [24] that provide overviews of security education, but they are currently out of date.

[2] MLS handles the government's need for multiple classification levels for information such as unclassified, confidential, secret and top secret.

2.2 Government Support for Academic Programs

The first NCISSE conference was held in 1997 and was established as a forum for dialog between government, industry and academia to define requirements for information security education and encourage development and expansion of information security curriculum at the graduate and undergraduate levels [6]. This conference was the earliest official forum to recognize computer security and bring together academics teaching security with key people in industry and government (Figure 1).

Around the same time 1996-1999, President Clinton established the President's Council on Critical Infostructure Protection (PCCIP) and subsequent President Decision Directive 63 (PDD63) [7]. In establishing the PCCIP, the president recognized the vulnerability of the US infrastructure and acknowledged its importance to national security. PDD63 simply expanded the definition of critical infrastructure to include cyber security [7].

Soon after PDD63 appeared, in 1999, NSA established the Centers for Academic Excellence in Information Assurance (CAEIA) [8]. These centers were academic institutions with expertise in cyber security as demonstrated by a number of security oriented faculty and curriculum that met federal security training standards [8]. The purpose of this program was to increase the number of "security professionals of different disciplines". During the first year, seven schools[3] were established and recognized at the Third Annual NCISSE conference [7].

In February of 2000, President Clinton released the National Plan for Information System Protection [9] which established the Scholarship for Service (SFS) program managed by the National Science Foundation (NSF) [10]. This program provided scholarships for undergraduate and graduate students for up to two years in exchange for an equal amount of Federal Job service upon graduation. In order for a school to obtain a scholarship program, they must first qualify as a Center for Academic Excellence [8]. During the first two years of the program, 150 students were enrolled [11].

In 2002, President Bush created the Department of Homeland Security (DHS) which united 22 agencies into one common group for the purpose of improving homeland security. One of their responsibilities was and continues to be funding R & D for new scientific understanding and technologies in support of Homeland Security [12].

In 2003, the President's National Strategy to Secure Cyberspace was passed by President Bush. It identified four major actions and initiatives for awareness, education and training which include [13]:

1. Promote awareness nationally to empower all Americans to secure their own systems.

2. Foster training and education programs to support national and cyber security needs.

3. Promote private sector to support widely recognized cyber security certification and training programs.

4. Increase efficiency of existing federal cyber security training programs.

[3]These schools were: James Madison University, George Mason University, Idaho State University, Iowa State University, Purdue University, University of California at Davis, and University of Idaho.

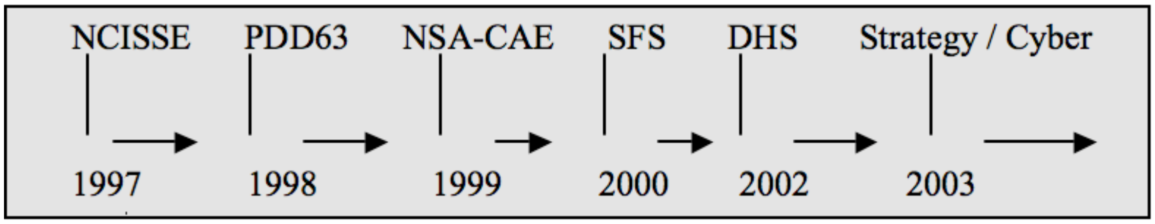


Figure 1. Timeline of Events Affecting Security Education

From 1997 to 2003, the US government created many initiatives for cyber security which appeared to recognize its importance for national security and the continued well-being of our nation. However, the only money allocated to academic security education was from the NSF SFS program.

3. COMPUTER SECURITY EDUCATION IN THE PRESENT

We now look at the present state of computer security education. We review the growing body of literature on established academic programs and discuss the typical approaches for establishing programs. Security education standards are discussed plus government and industry influences.

3.1 Academic Programs

Recently, a number of colleges have reported on their experiences adding computer security to their programs. Most schools appear to take one of two approaches: integrate security within individual CS courses [14, 15, 16] or create a special computer security degree or track[4] [17,18,19]. A few schools have opted for a combined approach where they have both specialized and integrated courses [20, 21]. There are reasons as to why a university chooses one approach over the other in their development of a computer security emphasis which often includes factors beyond the control of the institution. There are also tradeoffs with regards to these alternative approaches to computer security education.

Schools that choose to create a computer security track or special degree appear to have faculty that have experience in computer security or are strongly interested in pursuing security training [20, 19]. There also appears to be department or institutional support for a Security track and at least enough funding for course development. Integrating security into existing CS courses without offering specialized Security courses is one way that schools with limited resources in terms of faculty or funding can still offer security to students within their programs [14]. Faculty in these programs do not need not be retrained or develop completely new courses.

The effectiveness of each approach relates back to the goals for the CS graduates of a particular program in terms of computer security expertise. Programs that want to produce graduates with strong computer security skills capable of obtaining a computer security position have created specialized programs in computer security. CS programs that want their students to have exposure to computer security but not necessarily produce computer security professionals can achieve this through an integration of security principles into existing CS classes.

There is no clear evidence that specialized courses in computer security are superior to standard CS courses with integrated security components. However, schools that have chosen to integrate security within their existing curriculum point out several advantages over the specialized course path [14, 21]:

[4] Included are the schools that establish one or two specialized Computer Security courses

- Provide a security foundation to all their CS students as opposed to only those with a security interest
- Security concepts are learned within the broader CS topics such as system design, network administration, and programming
- The approach is available to all schools even those with limited resources and only requires faculty creativity and motivation

3.2 Government Support of Academic Programs

As discussed in Section 2.2, the government has several programs that currently support academic computer security education. Many of these programs were begun as the result of government initiatives related to national security. Here, we view the current status of these programs.

NSA-DHS Centers of Academic Excellence in Information Assurance [8]
Current: Has 67 Centers in 27 states
Started: 7 schools
Funding: Provides no monetary support for the Centers

NSF Scholarship for Service Program [10]
Current and Future: 350 graduates by 2005
Started: 150 enrolled
Funding: An annual budget of $30.5 million

The programs directly support Computer Security education within academic institutions. These programs appear to be thriving with an increasing number of Centers and students enrolled in scholarship programs.

3.2.1 Research Funding

In order to provide incentives for faculty and attract students into a field, the field of computer security needs a certain level of support in research dollars. Research fuels education by providing opportunities for faculty to publish, students to work on projects, and money to purchase equipment [23, 20]. Several long-time researchers and educators have noted that Computer Security needs a continued long-term commitment of basic research funding if it is to sustain itself as a viable area of study [22, 23].

At the first NCISSE conference in 1997, Bishop [23], a computer security researcher and educator, discussed the need for long-term funding as providing a stable base of resources and people which could be drawn from without having to continually start from scratch. In a later talk at CISSE[5] in 2000, Bishop commented that the government appears to be offering no support for basic security research which he states could ultimately prove disastrous [24].

Spafford briefed congressional staff in July, 2005 on the serious lack of funding in cyber-security research [22]. Spafford's group, the Presidential Information Technical Advisory Committee (PITAC)[6], issued a report in spring of 2005 that condemned the meager government investment in computer security research. Spafford noted that the NSF has become the primary agency for funding cyber security research with an annual budget of $30 million a year. This

[5] Between 1997 and 2000, NCISSE was changed to CISSE which is how it is currently known
[6] The group was disbanded in June of 2005

translates to only 8% of proposals being funded which as pointed out by the Computer Research Association (CRA) is discouraging student entry into the field [22].

The lack of long-term research funding was also noted by the Cyber Security Industry Alliance (CSIA), a group of security vendors who re-iterated PITAC findings in their own report [25]. The CSIA report stated that the Department of Homeland Security's budget in FY05 for science and technology is over $1 billion but that the budget for cyber security is just $18 million or about .02% [25].

Andy Birney, the editor of *Infosecurity* magazine, holds the government partly responsible for the nations' current cyber-security problems. Birney claims that a lack of government investment in security research discourages PhD students from entering the field [26]. This in turn creates shortages of faculty trained in security at academic institutions that produce the students entering the work force as programmers.

Another recent funding trend that affects computer security programs is the significant cuts from DARPA spending for university research [27]. DARPA has been a long-term source of basic Computer Science research funding for many years. This past year DARPA has cut the portion that goes to universities from $214 million to $123 million. They have shifted away from general research projects to more concrete deliverables produced in shorter time frames. This shift has resulted in a huge increase of proposals being directed towards NSF as one of the last Computer Science funding sources [27].

3.3 Industry View of Academic Programs

The computer industry comprises an important part of the United States economy, and almost all modern products and services use computer software. In the Report of the 2nd National Software Summit, leaders from academia, industry, and government argued that software should be elevated to an issue of national importance with a goal of "*Achieving the ability to routinely develop and deploy trustworthy software products and systems, while ensuring the continued competitiveness of the U.S. software industry"* [28]. The Build Security In (BSI) Software Assurance Initiative from the Department of Homeland Security seeks to achieve that goal in collaboration with academia and industry [29]. However, few companies accept responsibility for the poor quality of software that exists in most commercial products. Instead, some within the industry blame universities for producing programmers that don't know how to produce secure code.

Davidson, CSO of Oracle, appears to be a leading critic of academia [30, 31]. She believes academia should help shape the CS field and foster a culture of security. Davidson believes academia should produce CS majors that place more value on properties of safety, security and reliability above coolness and elegance. Davidson does not feel that industry should have to train programmers in security coding practices since they should have already acquired these skills prior to graduation [30].

Another group of software companies including Oracle and Microsoft, among others, discussed the failure of academic programs to produce security conscious programmers at a San Francisco Secure Software Forum in February, 2005 [31].

However, others point out that academia can't be entirely responsible for the problem of secure code. One panel member from the Secure Software Forum blames software companies that are still putting features above security [29]. The view that its partly industry's fault that we have so much bad software is supported by Birney as mentioned in Section 3.1. Birney refutes the popular belief that the root cause of vulnerabilities is insecure coding [26]. He discusses secure coding from three perspectives: academia, industry and government. Birney believes that a lack of government funding for academic computer security programs leads to a shortage of faculty with backgrounds in security as was discussed in the previous section. Furthermore Birney shifts some of the responsibility for vulnerable software to industry that still places development speed and profit over security.

While many in industry seem eager to blame academia for bad software without doing anything to help the situation, Microsoft is an exception in that they are working to fix the perceived problem. In 2002, Microsoft shut down for several weeks in order to train its workforce in secure software development [32]. Furthermore, they are one of the few companies investing in academic education through their 2-year old *Trustworthy Computing* curriculum program. They are offering $750,000 in grant money to 15 universities to produce security related curriculum. The curriculum materials are then made publicly available on their web site.

4. INSTITUTIONAL EXPERIENCES IN SECURITY PROGRAM DEVELOPMENT

In this section we offer the authors' individual experiences in Computer Security program development. Each program is different and is representative of various types of schools that develop security expertise in CS. The University of Idaho represents a mature, long-term security program since they were one of the original designated NSA CAE's. Indiana University of Pennsylvania is a more recently designated CAE (2002) and represents a less established security program. The computer security program at Northern Kentucky University is the smallest of the three and represents a non-CAE program that is mostly based on the efforts of a single faculty.

In establishing security programs at all three schools there were several commonalities noted. All three schools noted some difficulty with a lack of computer security curriculum standards. All programs began as the effort of one (or a few) faculty who instigated the security effort. All three schools are not major universities with large amounts of funding, so these programs were established in spite of limited funding. Students at the schools appear to be very interested in the topic and enrollment in the programs continues to be strong.

4.1 University of Idaho

Information assurance curriculum development at the University of Idaho began in 1991 with the arrival of Dr. Jim Alves-Foss. Dr. Alves-Foss graduated from UC Davis with a specialty in computer security and became the first IA faculty at the University of Idaho. The first security course developed consisted of a combined upper division undergraduate and graduate course, in computer security that emphasized both theory and practical knowledge. The addition of a second IA faculty, Dr. Debra Frincke, in 1993 resulted in the creation of several more security courses, Network Security and a senior/graduate level seminar in Intrusion Detection.

These early courses were followed by a senior/graduate course in Survivable System Analysis, a seminar in Security Policies and a course in Exploit Techniques and Defense. Other CS faculty became interested in IA and assisted with the development and teaching of these courses. In 2004, several additional courses were added including Forensics analysis and a lower level general Security Course [33]. These courses evolved as the perceived need arose and as an outgrowth of faculty research interests.

During the period of our curriculum development effort, we became an NSA CAE/IAE [8] and also participated in the NSF Scholarship for Service (SFS) program [10]. The NSA program has certain curriculum requirements which must be met in order to qualify for program continuance. The NSA CAE/IAE designation is closely tied to the National Security Telecommunications and Information Systems Security Committee's, NSTISSI[7] training standards especially 4011. In becoming an Academic Center of Excellence, the institution must

[7] In 2001, by Executive Order , NSTISSC was re-designated as CNSS, the Committee on National Security Systems.

demonstrate that their curriculum complies with the 4011 standard plus at least one other standard selected from the 4012 – 4015 documents [34]. Certification verifies that the college teaches skills that cover each of the seven topic areas of 4011.

In 2005, we have also begun integration of security concepts within several of our standard CS courses. We are planning on introducing secure coding into our beginner coding classes plus a computer security integrated software engineering class.

4.2 Indiana University of Pennsylvania

Indiana University of Pennsylvania (IUP) was designated a Centre of Academic Excellence in 2002. Since then, there has been noticeable improvement in curriculum development:

In 2003, a Bachelor of Information Assurance degree, jointly offered by the Computer Science department and the Criminology department track was introduced.

In 2005, a Master of Science in Information Assurance was recently developed. This is an interdisciplinary program designed to meet the industry and government needs for computer/network/information security professionals. The first offering will be in the fall of 2006.

– Through the NSF Cyber Security Capacity Building grant of 2001 – 2002, NSA Capacity grant of 2002- 2003, and the 2003 Cisco Equipment grant of $88,000, IUP established two isolated security laboratories, the Cyber Security and the Information Assurance laboratories, for teaching and research purposes.

– Through the SIGSCE Special Projects fund and local IUP Senate funding, hands-on exercises for Information Assurance courses have been developed. These are being pilot-tested in the department.

– To gain an industrial perspective of information assurance, industry partners provide guest lecturers on legal issues in Information Assurance classes and at the Information Assurance club meeting plus state police consultants provide guest lectures on legal issues.

One challenge in computer security education is the lack of body of knowledge for the computer security curriculum. During the summer of 2003 we started a project on augmenting and improving the teaching of the Cybersecurity Basics course at IUP. The Cybersecurity Basics course is an interdisciplinary course for the Criminology, Management Information Systems and Computer Science students. The course provides an introduction to the theories and concepts of computer security in host systems. The project involved 1) evaluating the effectiveness of host security tools in defending systems. 2) developing hands-on lab exercises based on the evaluated tools, and 3) integrating the developed hands-on lab exercises and the Cybersecurity theories and principles. Nine lab exercises were developed.

The development of teaching materials for Information Assurance courses can be a challenge. Most of the hands-on exercises required for such courses are based on tools for intrusion detection, forensic analysis, vulnerability analysis, firewall setting up, router auditing and packet sniffing. The challenge is that there is an abundance of CERT recommended security tools, tool version are changing often, and the teaching materials need to be continually updated.

4.3 Northern Kentucky University

Computer security curriculum development began at Northern Kentucky University (NKU) in 2002 with the introduction of a graduate computer security course by Dr. Charles Frank. Undergraduates enrolled in the class as a senior-level special topics elective course. The course focused on security fundamentals and network security and included a variety of lab exercises. The math department also offered a cryptology class, in which many computer science students enrolled.

In 2004, NKU added a new Computer Information Technology (CIT) degree with a track in Network, System Administration and Security. An undergraduate class in computer security was added as a requirement for the new track and as an elective for both CIT and CS majors.

NKU created a new College of Informatics in 2005, enhancing the ability of the departments of Computer Science and Information Systems to collaborate and hastening the pace of security curriculum development. Faculty designed a shared core curriculum for computer science, computer information technology, and information systems, and began mapping NKU's computer security curriculum to the CNSS 4011 standard as a preliminary step to becoming an NSA Center for Academic Excellence. The two departments will collaborate to offer a graduate certificate in Corporate Information Security in 2006.

The addition of a second faculty member, Dr. James Walden, with prior experience teaching computer security at the University of Toledo, helped the Department of Computer Science design new classes in Computer Forensics, Network Security, and Secure Software Engineering. The department is also beginning to integrate secure coding techniques into classes with a focus on programming. Building on the department's strengths in software development, a graduate certificate in Secure Software Engineering will be offered starting in 2006. Future plans include construction of a dedicated network security lab and development of a Master of Science degree program in Secure Software Engineering.

5. COMPUTER SECURITY EDUCATION IN THE FUTURE

So far, we have addressed the recent past of computer security education, *where we have been*, and the present, *where we are* with regards to programs, government and industry involvement. In this section, we discuss the future, *where we are going* with particular attention to objectives and potential barriers to success.

5.1 Computer Security Education Objectives

In trying to visualize the future of computer security education, it is useful to set goals and define specific objectives for reaching those goals. No one in the security field would argue with the general belief that providing a security background is beneficial to all students graduating from CS departments. One overall goal would be to increase the number of CS graduates with an understanding of computer security principles. Consequently, one objective that would help in reaching this goal is to increase the number of CS programs that teach computer security. Analyzing the specific steps needed to realize the objective of expanding the security education programs leads to a discussion of barriers to success for computer security education, the topic of the next section.

5.2 Barriers to Success for Computer Security

Achieving the objectives of promoting or increasing security concepts in CS programs requires some investment on the part of both institutions and the faculty member(s). These respective responsibilities for faculty and institutions are outlines in Table 1.

Table 1. Responsibilities for Promoting Computer Security

Responsibilities	
Institution	Faculty
Reduced Teaching Load	Learn Computer Security
Travel and/or Training Support Grants	Collaborate with Computer Sec. Research Inst.
Tenure Support	Travel to Conferences

In addition to the specific activities of faculty and institutions that wish to add computer security to the curriculum, there are other possible barriers to establishing a computer security emphasis. These are outside the control of faculty and their colleges and include:

- No standard for CS curriculum development
- Lack of government funding in basic research
- Limited industry involvement

Each of the barriers is explained in terms of its relations to computer security education.

There is currently no accepted standard for college level computer security curriculum development. This presents a barrier to the development of computer security programs. Without an accepted standard, departments must work harder to define course content [14, 19]. The lack of an academic computer security curriculum standard was recently studied by one of the authors [35]. That study noted the inadequacy of the 40XX Training Standards for academic programs and described the problems faced by academia in trying to map their curriculum to these standards.

The lack of government funding was addressed in a previous section and noted as a disincentive for promotion of security education. If there is little or no research funding available in a given field of study, then there is no way to support graduate students who are to become future faculty and eventually promote their own research programs. Consequently, disciplines that lack research support struggle to recruit students and faculty since there is a perception of a lack of resources in the field. The current dismal situation where only a small percentage of cyber security proposals are funded by the NSF is not conducive to promoting computer security programs within academia.

The software industry is concerned with the perceived lack of security awareness in students graduating from CS programs. Yet, they are not as a group volunteering to assist with this problem either by funding or other direct involvement. The objectives of the CISSE conference were to establish a working partnership between government, industry and academia [6]. Industry and government should provide better support for higher education. Yet, outside of the institutions with large, well-established programs, partnerships between industry and academia are not common. In addition to directly funding academic research projects, industry could provide a number of other opportunities. There could be an exchange of faculty and industry in internship settings in order to share expertise, students could benefit by working on real problems [23], industry could serve on Department advisory boards.

6. CONCLUSION AND FUTURE WORK

In this paper, we have provided an overview of computer security education. We presented government initiatives and other events from the past eight years, examined the current state of academic progress and discussed future objectives for promoting security within CS along with the perceived barriers to success. There are a number of areas that need to be addressed in order for security education to progress. A lack of research funding in academic programs appears to be a major roadblock to the creation of a viable national security program. Faculty training along with institutional support appears to be a problem for programs that lack any faculty with a background in security.

6.1 Future Work

There is a strong need for a survey of CS and IT departments to determine current status, future plans and needs for security education. The University of Idaho is planning to survey schools that have mapped their curriculum to the CNSS[8] 40XX training standards to get feedback on their experience mapping their curriculum to the 40XX criteria. However this is intended to be a targeted survey and not a comprehensive survey of all CS departments.

Other projects that would benefit security education include:

– An academic curriculum standard for both undergraduate and graduate programs

– Integration of computer security into accreditation programs (e.g. ABET)

– Support for schools beginning security programs

  – Curriculum help and mentorship from established programs

Those who work in both security and education see promotion of security education for all graduating CS majors as one of the few concrete steps we can take that could favorably improve the state of cyber security. Ultimately, security education should increase the level of competence in our developers to produce better quality, more robust systems capable of surviving most disruptions, intentional or otherwise.

REFERENCES

1. Saita, A. "Ripe for Harvest", Information Security, Vol. 8.3, March 2005.
2. Dittrich, D. "Invasion Force", Information Security, Vol. 8.3, March 2005.
3. Landwehr, C. E., "Changing the Puzzle Pieces", IEEE Security & Privacy, Vol. 3,1, Jan/Feb 2005
4. Schell, R. "Information Security: Science, Pseudoscience and Flying Pigs", Invited Essay, ACSAC 2001.
5. Spafford, E. "A Failure to Learn from the Past", ACSAC, 2003, http://www.acsac.org/2003/papers/classic-spafford2.pdf
6. CISSE. "Colloquium for Information Systems Security Education", http://www.nisse.org
7. PCCIP and PDD63. http://www.ciao.gov/PCCIP/report_index.html
8. NSA. "NSA Centers for Academic Excellence", http://www.nsa.gov/isso/programs/nietp/newspg1.htm
9. National Plan for Information System Protection. http://www.gao.gov/new.items/d02474.pdf
10. National Science Foundation Scholarship for Service. http://www.sfs.opm.gov
11. NSF SFS Statistics, http://www.internetnews.colm/bus-news/article.php/1585651
12. Department of Homeland Security. Announcement. http://www.dhs.gov/dhspublic/display?theme=10
13. Null, L. "Integrating Security Across the Computer Science Curriculum", Consortium for Computing Sciences in Colleges, (CCSC) 2004.
14. Irvine, C.E., Chin, S. and Frincke, D., "Integrating Security into the Curriculum", IEEE Computer, pp. 25-30, December 1998
15. Mullins, P. et. al. "Panel on Integrating Security Concepts into Existing Computer Courses", SIGCSE '02, Feb. 27 Mar. 3, 2002, Covington, KY, 2002.

[8] In 2001, the President re-designated the National Security Telecommunications and Information Systems Security Committee (NSTISSC) as the Committee on National Security Systems (CNSS) which resulted in a renaming of the training standards to CNSS instead of NSTISSC

16. Werner, L. "Teaching Principled and Practical Information Security", Consortium for Computing Sciences in Colleges, (CCSC) 2004.
17. Azadegans, S. et. al. "An Undergraduate Track in Computer Security", ACM SigCSE Bulletin, Vol. 35, 3, June 2003.
18. Crowley, E. "Information System Security Curricula Development", Proceed. of Conf. on Inform. Tech. Curriculum (CITCA '04), Oct. 16-18, 2003, Lafayette, IN.
19. Vaughn, R. Jr., Dampier, D. and Warkentin, M. B. "Building an Information Security Education Program", InfoSecCD Conference '04, Oct. 8, 2004, Kennesaw, GA
20. Yang, A. "Computer Security and Impact on Computer Science Education", Jr. of Computing in Small Colleges, Vol. 16, 4, April, 2001.
21. Computer Research Association Policy Blog. http://www.cra.org/govaffairs/blog/index.php
22. Bishop, M. The State of INFOSEC Education in Academia: Present and Future Directions, keynote speech, NCISSE, pp 19-33, Apr. 1997.
23. Bishop, M. "Computer Security Education: Training, Scholarship and Research", keynote speech, CISSE, 2000.
24. CSIA. "Federal Funding for Cyber Security R & D", CSIA Alliance, July 2005, http://www.csialliance.org/CSIA_RD.pdf
25. Birney, A. "Secure Coding? BAH!", Editorial, InfoSecurity, Jan. 2004.
26. Markoff, J. "Failure of Federal Cyber Security Funding", New York Times, June 2005.
27. CNSS, Report of the 2nd National Software Summit, http://www.cnsoftware.org/nss2report/NSS2FinalReport04-29-05PDF.pdf, April 29, 2005
28. Gary McGraw and Nancy Mead, "Engineering Security Into the Software Development Life Cycle ," Crosstalk: The Journal of Defense Software Engineering, October 2005, http://www.stsc.hill.af.mil/crosstalk/2005/10/0510McGrawMead.html
29. Davidson, M.A. "Leading by Example: the case for IT Security in Academia", Educause Review, Jan/Feb. 2005.
30. Lemos, R. "Software Firms Fault Colleges' Security Education", c/net News.com, Feb. 2005, http://news.com.com/Software+firms+fault+colleges+security+education/2100-1002_3-5579014.html
31. Microsoft Research. "Microsoft Curriculum Grants", http://www.microsoft.com/presspass/press/2005/jul05/07-18FacultySummit/05PR.mspx
32. CSDS. "CS Based IA Curriculum", http://www.csds.uidaho.edu/IA/IAStudy.htm.
33. NSA. "National IA Education and Training Program", http://www.nsa.gov/ia/academia/cnstesstandards.cfm
34. Taylor, C. and Alves-Foss, J. "The Need for Information Assurance Curriculum Standards", Proceedings of 2005 CISSE, June 6-9, Atlanta, GA, 2005.

AN ANIMATED SIMULATOR FOR PACKET SNIFFER

Xiaohong Yuan, Percy Vega, Jinsheng Xu, Huiming Yu, and Stephen Providence

Department of Computer Science, North Carolina A&T State University, 1601 East Market St., Greensboro, NC 27411, Email: {xhyuan, jx, ucshmyu, svp} @ncat.edu; Phone: (336)3347245. [*]Everett Consulting Corp., 320 McKinley St. #10, Hollywood, FL 33019, Email: percyvega@gmail.com Phone: (954)7325664

Abstract: Visualization and animation have been used to graphically illustrate various concepts in computer science. This paper describes an animated simulator for packet sniffer. The goal of this tool is to provide users with interactive tutorials and simulations to help them better understand the security concept packet sniffer and related computer networks concepts. This tool can be used in an introductory level computer security course or a computer networks course. It can be used by the instructor in the classroom, and can also be used by the students outside the classroom. This tool will be used in our institute and its effectiveness in teaching and learning will be assessed.

Keywords: Computer science education, security, packet sniffer, animation

1. INTRODUCTION

Visualization has been used in computer science education in the fields of algorithm, computer networks, computer architecture, and operating systems [1-4]. The surveys of computer science educators conducted by the Working Group on "Improving the Educational Impact of Algorithm Visualization" suggest a widespread belief that visualization technology positively impacts learning [5]. It seemed that computing educators are convinced that visualization can make a difference in helping learners better learn the concepts.

It has become increasingly important to provide today's computing students with training and education in security issues. Universities have tried to incorporate security issues into computer science curriculum. In some computer science programs, some security topics are incorporated into existing courses. Others developed new, required courses to the computer science curricula [6, 7, 8]. Resources for teaching computer and information systems security at the undergraduate level will be useful for computer science educators [9].

This paper presents a software tool that demonstrates the security concept of packet sniffer and related computer networks concepts through animation. It is intended to be used in an undergraduate level computer security course or a computer network course. To make this tool more beneficial, exercise problems are designed to improve learner's involvement in learning with this tool [10]. We believe this visualization tool will help students better understand the packet sniffer and network concepts, improve student participation, and make teaching and learning more enjoyable.

2. PACKET SNIFFER

Packet sniffer is a program that captures all of the data packets that pass through a given network interface, and recognizes and decodes certain packets of interest. A packet sniffer is sometimes referred to as a network monitor, or network analyzer. It is normally used by network or system administrator to monitor and troubleshoot network traffic. However, it is sometimes also used by malicious intruders for illicit purpose such as stealing a user's password or credit-card number [11].

Typically, a computer in a network would only capture data packets that are intended for it and ignore packets that are not intended for it. However, if its network interface is configured into promiscuous mode, the computer is capable of capturing all packets traversing the network regardless of the destinations of the packets. A packet sniffer can only capture packets within a given subnet.

There exist various commercial and free packet sniffer tools. Ethereal is an open source tool for troubleshooting, analysis, software and protocol development and education. It can capture data from a live network connection, or read from a capture file. Captured network data can be browsed via a GUI, and more than seven hundred protocols can currently be dissected. A display filter is provided to refine data display [12]. AnalogX PacketMon is a fast and simple to use network monitor that captures IP packets. It uses advanced rules for filter [13]. Network Probe [14] is a tool for traffic-level network monitoring, analysis and visualization. It provides full graphical representation and detailed statistics of the traffic and the type of data traveling across the network. It allows the user to isolate traffic problems and congestions. It also allows the user to search, sort, and filter network traffic information by protocols, hosts, network interfaces, etc. There are yet many other similar packet sniffer tools available.

3. THE PACKET SNIFFER SIMULATOR

The packet sniffer simulator demonstrates visually how a packet sniffer works in a local area network environment, and how data packets are encapsulated and interpreted while going through the protocol stack. The packet sniffer simulator consists of a suite of five demos. Demo I to IV progressively demonstrate how a packet sniffer works. The simulation is based on a network with two subnets. The two subnets are connected with a router, and each subnet has a hub. The first subnet has a star topology and the second subnet has a bus topology. Demo V shows a protocol stack and how a data packet is encapsulated while going down the protocol stack at the source computer, and interpreted while going up the protocol stack at the destination computer.

The learning objectives of this packet sniffer simulator are:

a) Explain the differences between a hub, a bridge/switch, and a router

b) Explain bus and star topology

c) Explain how a data packet is transmitted in a local area network

d) Explain the purpose of "promiscuous mode" of a network interface

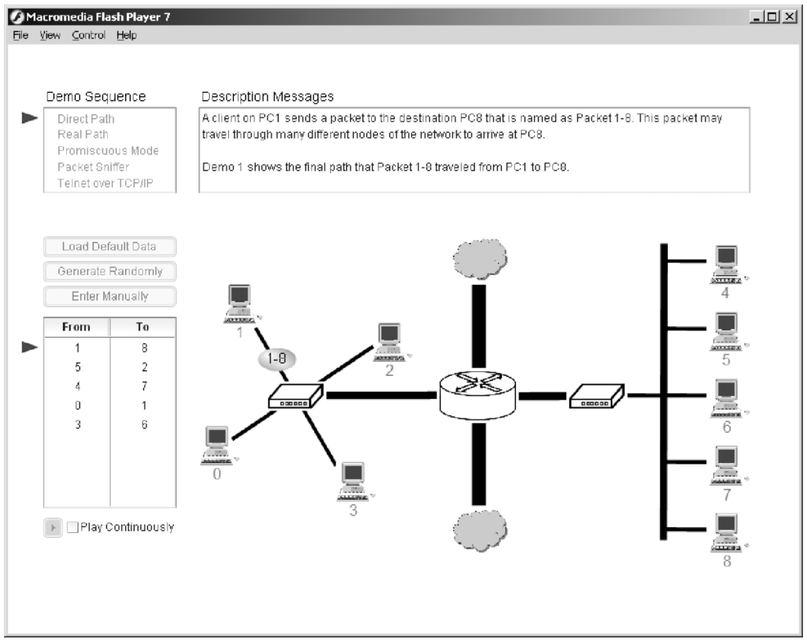
e) Explain what a packet sniffer does, and how it works.

f) Explain the encapsulation and de-encapsulation process of a data packet while going through the protocol stack

Macromedia Flash MX Professional Edition was selected to implement the demos of the packet sniffer simulator. The main environment in which Flash animations can run is web page, as a Flash Applet. These animations can also run as a standalone application. The only requirement for both environments is to download and install Macromedia Flash Player which is freely available. The following sections explain the five demos, and the network and security concepts they demonstrate.

3.1 Demo I: Direct Path

Figure 1 shows the user interface of Demo I. Demo II, III, and IV have the same user interface as Figure 1. The user interface can be grouped into four components. The top-left component "Demo Sequence", is a textbox that lists the names of the five demos. It allows the user to select the demo he wants to view. The top-right component "Description Messages", is a text-area that briefly describes the animation. The bottom-right component shows the network architecture. It shows two subnets connected by a router. The subnet to the left has a star topology; the subnet to the right has a bus topology. Each subnet has a hub connecting the computers. The computers are numbered from 0 to 8. The bottom-left component displays the data for simulation. The data used in the simulation is the source address and the destination address of a data packet. Three buttons are provided, which allow user to select from loading default data from an input file, or generating input data randomly, or entering data manually. The input data are displayed in the table that has two columns. Under the first column "From" are source addresses of the data packets. Under the second column "To" are destination addresses of the data packets. Under the two-column data table, are a "play" button (an arrow within a square), and a checkbox "Play continuously". If the checkbox "Play continuously" is checked and the "play" button is clicked, the animation will run from Demo I through Demo IV sequentially and go through all the data pairs in the data table sequentially in each demo. If the "Play continuously" button is not checked and the "play" button is clicked, it will only show one step of the animation, i.e., the animation for one data set. The user can step through the simulation by repeatedly clicking the "play" button. At each step of the simulation, the "description messages" text-area is updated to reflect what is going on.



A client on PC1 sends a packet to the destination PC8 that is named as Packet 1-8. This packet may travel through many different nodes of the network to arrive at PC8.

Demo 1 shows the final path that Packet 1-8 traveled from PC1 to PC8.

From	To
1	8
5	2
4	7
0	1
3	6

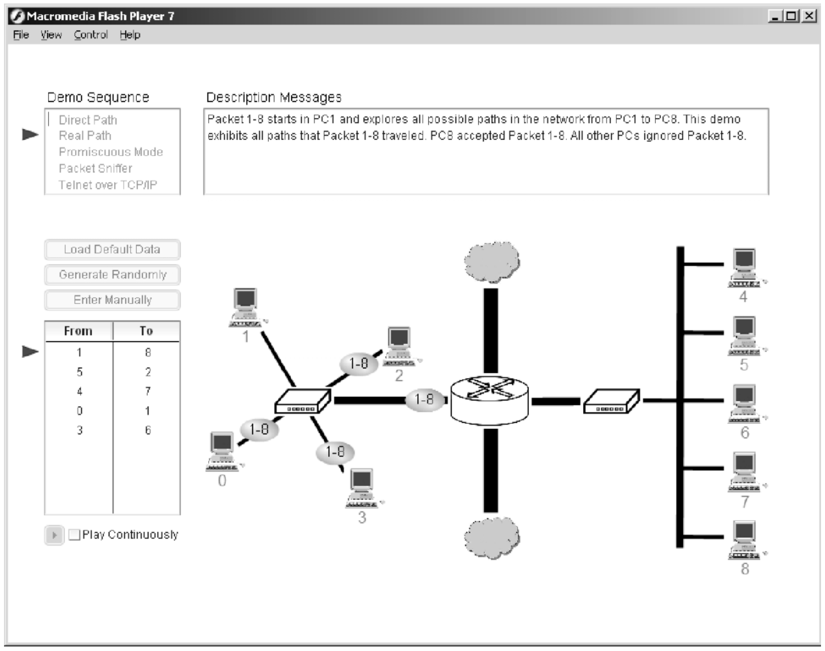
Figure 1. Demo I: Direct Path

Demo I displays the path a data packet from a source goes through to reach the destination. A data packet is represented as an oval shape, labeled by the source and destination numbers. For example, Figure 1 shows a data packet, Packet 1-8 moving from computer 1 to the hub. The packet will go through the hub to the left, the router in the middle, the hub to the right, and finally arrive at computer 8. Keep in mind that, since hub is used in the two subnets, the real path that Packet 1-8 traversed is not the same as the direct path. Demo II demonstrates the real path of a data packet in the simulated network.

3.2 Demo II: The Real Path

Since the computers in the subnet to the left are connected by a hub, all traffic can be seen by all computers on the subnet. The same is true with the subnet to the right which has the bus topology. A network segment that is not switched or bridged (i.e., connected through a hub) is called a common collision domain. Physically, any signal sent across a common collision domain reaches all attached computers. The network interface hardware detects the electrical signal and extracts a copy of the frame. It checks the address of each incoming frame to determine whether it should accept the frame. The network interface hardware compares the destination address in the frame to the computer's physical address (also called hardware address, or media access address). If the destination address in the frame matches the computer's physical address, the interface hardware accepts the frame and passes it to the operating system. If the destination address in the frame does not match the computer's physical address, the interface hardware discards the frame and waits for the next frame to appear [15].

Figure 2 shows that, when computer 1 generates a data packet, the data packet is captured by computer 0, 2 and 3 in the same subnet. Meanwhile, the data packet is forwarded to the router in the middle of the network. The data packet is forwarded to the subnet to the right by the router, and all the computers attached to the bus receive Packet 1-8. Only computer 8's network interface accepts Packet 1-8, and the other computers discard Packet 1-8. This is depicted in Figure 3, in which the comment "mine" pointing to computer 8 indicates that the data packet is accepted by computer 8's network interface, and the comment "not mine" indicates the data packet is discarded by the computer's network interface.



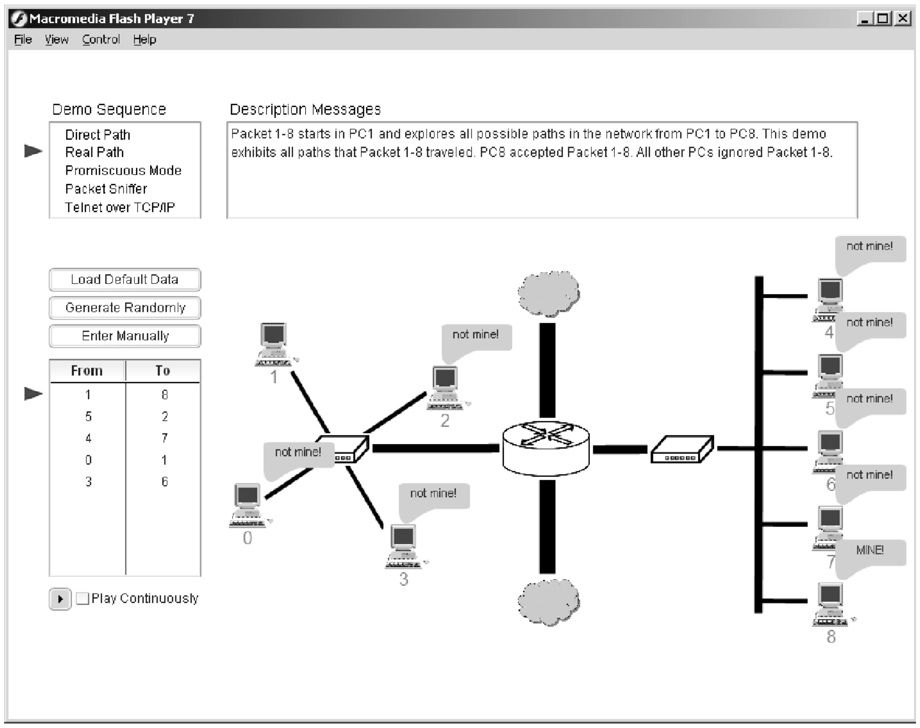
Description Messages

Packet 1-8 starts in PC1 and explores all possible paths in the network from PC1 to PC8. This demo exhibits all paths that Packet 1-8 traveled. PC8 accepted Packet 1-8. All other PCs ignored Packet 1-8.

From	To
1	8
5	2
4	7
0	1
3	6

Figure 2. Demo II (a): The Real Path When a Packet is Sent by the Source Computer

If the hub in the subnet to the left is replaced by a switch or bridge, and the computers in the subnet to the right are connected to a switch or bridge too, then the path Packet 1-8 traverses will be the same as Demo I. By comparing Demo I and Demo II, the concepts of repeater, hub, bridge, switch and router can be reviewed.



Packet 1-8 starts in PC1 and explores all possible paths in the network from PC1 to PC8. This demo exhibits all paths that Packet 1-8 traveled. PC8 accepted Packet 1-8. All other PCs ignored Packet 1-8.

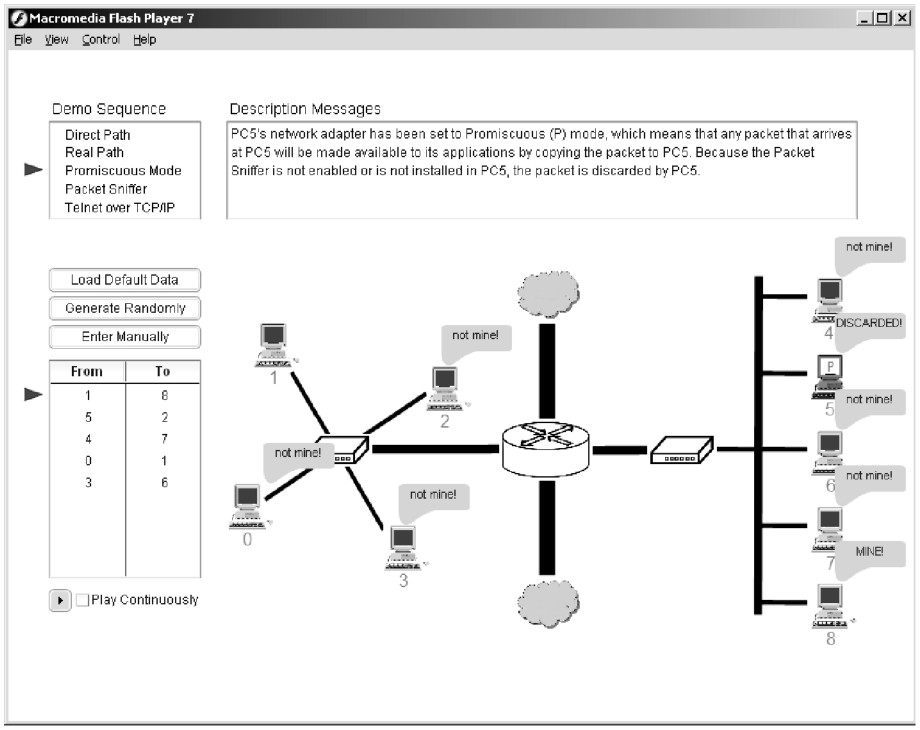
From	To
1	8
5	2
4	7
0	1
3	6

Figure 3. Demo II (b): The Real Path When a Packet is Received by a Destination

A repeater is a hardware device used to extend a network cable. It has two ports. A repeater receives signal from one port, regenerates the signal and sends out to the other port. A hub is a multi-port repeater. A hub receives signal on one port, and regenerates the signal and sends the signal out to all the other ports. Repeaters and hubs work at the Physical layer of the OSI model. A bridge works at the data-link layer of the OSI model. It examines each incoming packet. First the media access address (MAC address) of the sender and the port number through which the packet enters are added into the routing table of the bridge. Then the MAC address of the recipient is looked up from the routing table to determine which port should the packet be forwarded to. If the recipient's MAC address is in the routing table, then the port number is looked up and the packet is forwarded to that port. If the recipient's MAC address is not in the routing table, the bridge sends the signal to all ports except for the one where it was received. A switch is a multi-port bridge. It has the functions of a bridge, but uses a dedicated processor to implement the function, so it is faster than traditional software based bridges. A router works at the network layer of the OSI model. It uses network addresses (for example, IP address) to determine how to forward a packet [16].

3.3 Demo III: Promiscuous Mode

Normally, a computer's network interface hardware checks the destination address of each incoming frame to determine whether it should accept the frame. It discards a frame whose destination address does not match its physical address. However, a computer's network interface hardware can be configured by software into promiscuous mode, which overrides the conventional address recognition. Once in promiscuous mode, the network interface does not check the destination address, but accepts all frames. The network interface simply places a copy of each frame in the computer's memory and informs the CPU about the arrival of the frame [15].



PC5's network adapter has been set to Promiscuous (P) mode, which means that any packet that arrives at PC5 will be made available to its applications by copying the packet to PC5. Because the Packet Sniffer is not enabled or is not installed in PC5, the packet is discarded by PC5.

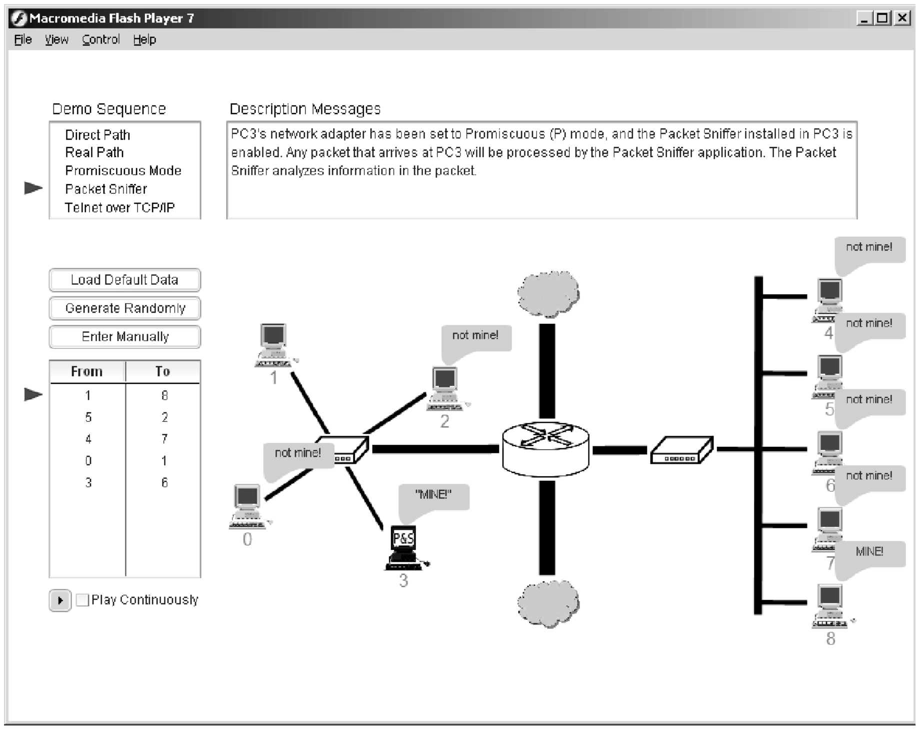
From	To
1	8
5	2
4	7
0	1
3	6

Figure 4. Demo III: Promiscuous Mode

Figure 4 shows the result of an animation of data packet transmission with Computer 5 configured into promiscuous mode. When Packet 1-8 was transmitted on the common bus in the subnet to the right, computer 5 accepted the packet even though the packet was not addressed to it. However, since there was no packet sniffer installed on computer 5 to process the packet, the packet was simply discarded by the operating system of computer 5. Notice the difference between the "DISCARD" comment and the "not mine" comment. In the case of "not mine", the computer's network interface hardware discards the data packet because the destination address is not the same as the physical address of the computer; whereas in the case of "DISCARD", the data packet is accepted by the network interface hardware but is discarded by the operating systems of the computer.

3.4 Demo IV: Packet Sniffer

Figure 5 shows the result of an animation when computer 3 has a packet sniffer installed. The network interface on computer 3 is also configured into promiscuous mode. When Packet 1-8 was sent out to the network, computer 3 captured the package and accepted it, even though the packet was not addressed to computer 3. This is indicated by the "mine" comment in Figure 5. The packet sniffer then examines the content of the data packet according to its configuration. The packet sniffer can be configured by the user to determine what fields to examine and what information it keeps. For example, a packet sniffer can be configured to examine all frames originated from a particular computer, or all TCP/IP traffic, or gather general traffic statistics.



Demo Sequence

- Direct Path
- Real Path
- Promiscuous Mode
- ► Packet Sniffer
- Telnet over TCP/IP

Description Messages

PC3's network adapter has been set to Promiscuous (P) mode, and the Packet Sniffer installed in PC3 is enabled. Any packet that arrives at PC3 will be processed by the Packet Sniffer application. The Packet Sniffer analyzes information in the packet.

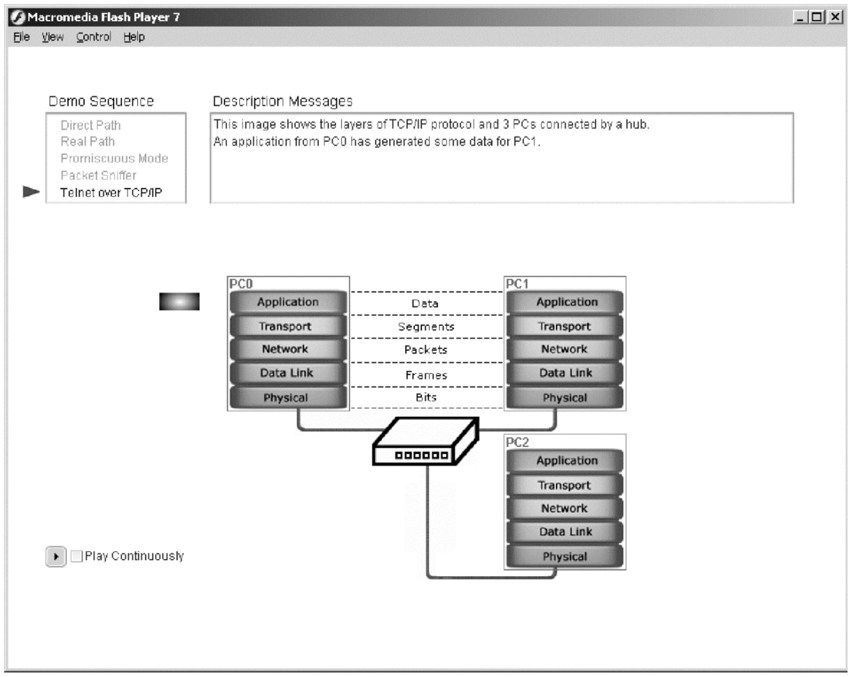
	From	To
►	1	8
	5	2
	4	7
	0	1
	3	6

Figure 5. Demo IV: Packet Sniffer

Consider a packet sent from computer 4 to computer 7. When Packet 4-7 reaches the router, it will not be forwarded to the subnet to the left, so computer 3, which has the packet sniffer installed on it, will not be able to capture Packet 4-7. Similarly, if the packet sniffer is installed in one of the computers in the subnet to the right, it would not be able to capture data traffic going through the subnet to the left. A packet sniffer only works in a common collision domain.

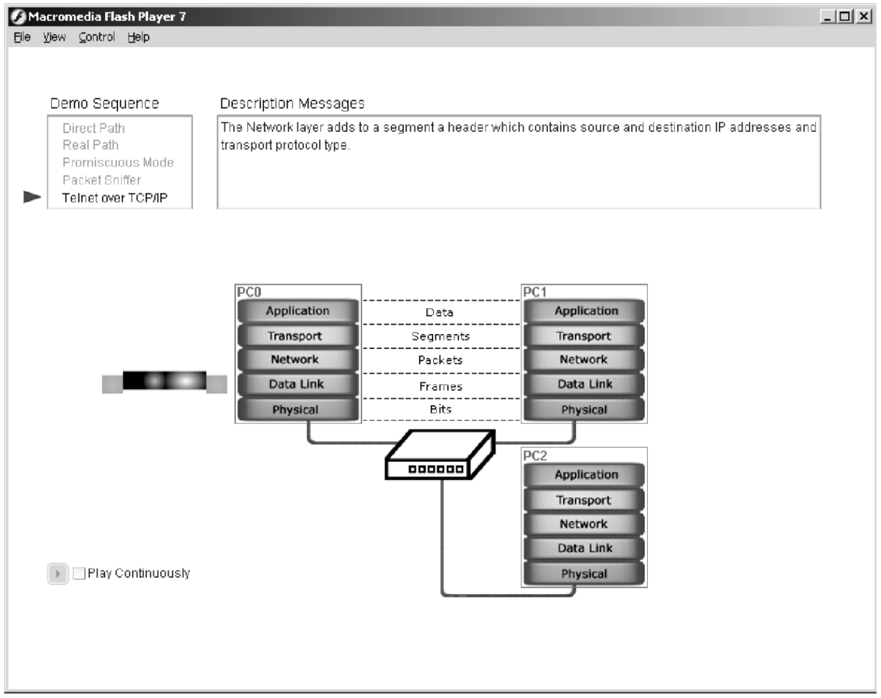
3.5 Demo V: Telnet Over TCP/IP

Demo V demonstrates how a data packet is encapsulated and de-encapsulated while going through the protocol stack. It assumes a Telnet application sending data packets over a network with TCP/IP protocol. Figure 6 represents three computers (PC0, PC1 and PC2) connected to a hub. Each computer is represented by a rectangle. In each rectangle a protocol stack of five layers are displayed. The five layers are: application, transport, network, data link and physical layer. The animation shows a data packet generated at the application layer at PC0 being encapsulated while moving down through the protocol stack, and being de-encapsulated while moving up through the protocol stack at PC1 (Figure 7 and Figure 8). The user can step through the animation by clicking the "play" button repeatedly, or run the simulation continuously by checking the checkbox "Play Continuously".



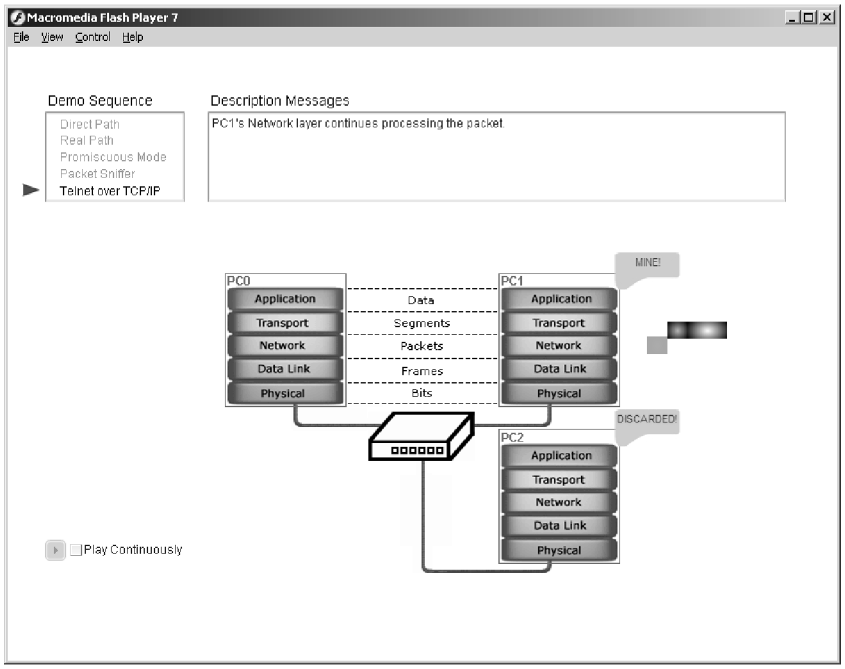
This image shows the layers of TCP/IP protocol and 3 PCs connected by a hub.
An application from PC0 has generated some data for PC1.

Figure 6. Demo V (a): A Data Packet Generated by PC 0 at Application Layer



The Network layer adds to a segment a header which contains source and destination IP addresses and transport protocol type.

Figure 7. Demo V (b): The Encapsulation Process



PC1's Network layer continues processing the packet.

Figure 8. Demo V (c): The De-Encapsulation Process

Encapsulation means that, when the data packet moves down through the protocol layers, a header (and a trailer sometimes) is added at each protocol layer. The transport layer adds a header that contains the source and destination port numbers. The network layer adds a header that contains the source and destination IP addresses and the transport protocol type. The data link layer adds a header and a trailer with source and destination physical addresses and the network type. And the physical layer converts the frame generated by the data link layer into bits. On the destination computer, the de-encapsulation process occurs. The headers (or trailers) are removed in reverse order.

In Figure 8, the data frame is transmitted to PC1 and PC2. At the data link layer, the network interface hardware of PC2 recognizes the destination address of the incoming frame, and finds out it is not addressed to PC2, so PC2 discards the data packet at the data link layer. Whereas PC1's network interface hardware recognizes that the data packet is addressed to PC1. So the frame is forwarded to the operating systems and is further de-encapsulated until it reaches the application layer.

4. RELATED WORK

Holliday [2, 17] developed a series of Java applets and explanatory material to illustrate key computer networking concepts. The applets are: protocol stack applet, error-control applet, reliable data transfer applet and media access applet. The Demo V of our packet sniffer simulator is similar to the protocol stack applet. The protocol stack applet demonstrates how a message goes from the source machine to the destination machine across a router. Whereas Demo V shows how a message goes from the source machine to the destination machine and other machines connected to a hub. Both demonstrate the encapsulation/de-encapsulation process. The protocol stack applet allows the user to specify the sizes of the message and the headers at the different protocol stack layers. The Demo V allows the user to step through the simulation or run the simulation continuously.

White [18] developed a set of visualization tools for teaching a data communications and computer networks course. A collection of eleven computer based training modules was created as an educational supplement for a textbook on data communications and computer networks. Some of the modules demonstrate similar networking concepts as our packet sniffer simulator. For example, Module 1 demonstrates the concept of encapsulation and de-encapsulation; Module 6, 7 and 8 demonstrates the basic concepts of local area networks, and simulate internetworking with hubs, bridges and switches.

The "increasing security in aviation-oriented computing education: a modular approach" project at Embry-Riddle Aeronautical University [19, 20] is developing interactive modules for several topics: buffer overflow vulnerabilities, cryptography, and multimodal transportation and Bioterrorism defense. These interactive modules use software simulation and visualization to demonstrate security concepts. They may be used by an instructor for classroom or laboratory work. Our packet sniffer simulator can be considered as an interactive module with similar intention, demonstrating the packet sniffing vulnerability in a network environment.

CyberCIEGE [21] is a high-end, commercial-quality video game developed for teaching security concepts and practices. It is a resource-management simulation in which the players engage in planning and construction and observe the results of their choices.

5. CONCLUSION AND FUTURE WORK

The packet sniffer animator demonstrates the packet sniffer and local area network concepts through five demos. It can be used by instructors of computer networks and security in the classroom. Tutorials and exercises are designed with the tool to help students to gain deep understanding of packet sniffer and related network concepts. In [5], six forms of learner engagement with visualization technology are defined, they are: 1) no viewing; 2) viewing; 3) Responding; 4) Changing; 5) Constructing; and 6) Presenting. We can design exercises so that the students can be engaged in the forms of responding and changing. The packet sniffer animator can be accessed at:

http://clayton.ncat.edu/comp476/PacketSnifferAnimation/index.html

We plan to use this tool in an undergraduate level course Computer Networks, and an Applied Network Security course in the Fall 2005 semester. The effectiveness of the tool in teaching and learning will be assessed in these two courses. Questionnaire will be designed to collect students' opinion on this tool. The future work will also include the development of animation tools for other more complicated security concepts, for example, the Kerberos architecture and distributed denial of services, and the assessment of these tools.

The CAPE and eLMS [22] developed at Vanderbilt University's Institute for Software Integrated Systems are a technology infrastructure for adaptive on-line learning. CAPE is used to design how learning materials are used to create an adaptive learning experience, and eLMS is a web-based delivery platform. We are interested in investigating the possibility of using CAPE and eLMS to extend the Packet Sniffer simulator to create adaptive learning experiences for computer network and computer security courses.

ACKNOWLEDGEMENT

This work was supported by National Science Foundation under the award number DUE-0415571.

REFERENCES

1. Harrold, M. J. and Stasko, J. Algorithm animation, 2002.Available at: http://www.cc.gatech.edu/gvu/softviz/algoanim/
2. Holliday, M. A. Animation of computer networking concepts, ACM Journal of Educational Resources in Computing, Vol. 3, No. 2, June 2003, Article 2.
3. Null, L. and Rao, K. CAMERA: Introducing memory concepts via visualization, Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education, St. Louis, Missouri, USA, February 23-27, 2005, pg. 96-100.
4. Carr, S., Mayo, J. and Shene, C.K. ThreadMentor – A system for teaching multithreaded programming, 2003. Available at: http://www.cs.mtu.edu/~shene/NSF-3/
5. Naps, T. L. et. al. Exploring the role of visualization and engagement in computer science education, ACM SIGCSE Bulletin, Vol. 35, Issue 2, 2003, pg. 131-152.
6. Frincke, D. and Bishop M. Joining the security education community, IEEE Security and Privacy, Vol. 2, Issue 5, 2004, pg. 61-63.
7. LeBlanc, C. and Stiller E. Teaching computer security at a small college. Proceedings of the 35th SIGCSE Technical Symposium on Computer Science Education, Norfolk, Virginia, USA, March 3-7, 2004, pg. 407-411.
8. Mullins, P. et. al. Panel on integrating security concepts into existing computer courses, Proceedings of the 33th SIGCSE Technical Symposium on Computer Science Education, Nrothern Kentucky, Cincinnati, USA, February 27-March 3, 2002, pg. 365-366.
9. Bhagyavati, et. al. Teaching hands-on computer and information systems security despite limited resources, Proceedings of the 36th SIGCSE Technical Symposium on Computer Science Education, St. Louis, Missouri, USA, February 23-27, 2005, pg. 325-326.
10. Grissom, S. et. al. Algorithm visualization in CS Education: Comparing levels of student engagement, Proceedings of ACM 2003 Symposium on Software Visualization, San Diego, California, , USA, June 11-13, pg. 87- 94.
11. Bradley, etc. "Introduction to Packet Sniffing", 2005. Available at: http://netsecurity.about.com/cs/hackertools/a/aa121403.htm
12. Combs, G. Ethereal, 2005. Available at: http://www.ethereal.com/
13. AnalogX, PacketMon, 1998-2003. Available at: http://www.analogx.com/contents/download/network/pmon.htm
14. Objectplanet, Network Probe, 2005. Available at: http://www.objectplanet.com/probe/
15. Comer, D. E. Computer Networks and Internets, 4th edition, Pearson Prentice Hall, 2004.
16. Shields, P. Networks in-depth: switches, hubs and routers, 2005. Available at: http://www.thebusinessmac.com/features/network_indepth.shtml
17. Holliday, M. A. and Johnson, M. "A Web-Based Introduction to Computer Networks for Non-Majors - The Protocol Stack", February, 2004. Available at: http://cs.wcu.edu/~holliday/cware/Stack/indexStack.html
18. White, C. M. Visualization tools to support data communications and computer network courses", Journal of Computing Sciences in Colleges, Vol. 17, Issue 1, pg. 81-89.
19. Crandall, J.R., et. al. Driving home the buffer overflow problem: a training module for programmers and managers, Proceedings of National Colloquium for Information Systems Security Education (NCISSE 2002), Washington, 2002.
20. Gerhart, S. et. al. Increasing security in aviation-oriented computing education: a modular approach, August 2005. Available at: http://nsfsecurity.pr.erau.edu/
21. Irvine, C. E. and Thompson, M. F. "CyberCIEGE: Gaming for Information Assurance", IEEE Security and Privacy, Volume 3, Issue 3, pg. 61-64.
22. Sztipanovits, J. et al. "Introducing Embedded Software and Systems Education and Advanced Learning Technology in Engineering Curriculum", ACM Transactions of Embedded Computing Systems, Special Issue on Education, 2005.

THE SCIENCE OF INFORMATION PROTECTION

Daniel F. Warren

Naval Postgraduate School

Abstract: The presentation of Information Protection material can be improved in two important ways. First, if the material is arranged in a systematic/scientific fashion it can show how all the various pieces fit together and it can also demonstrate completeness by showing that all threats are addressed. Second, if each protection technique is preceded by a clear description of the threat that it addresses learning is significantly enhanced because the protection technique is motivated. This paper presents an information threat model that 1) arranges the material in a scientific/systematic fashion and 2) facilitates a threat-first presentation of Information Protection techniques.

Key words: Information Protection, Access Control

1. INTRODUCTION

The threat model presented here is based on the following seven threat classes. These threat classes are mutually disjoint and together constitute all threats.

1. The Non-Human threat
2. The Human Error threat
3. The Authorized Person threat
4. The Unauthorized Insider threat
5. The Outsider Attacking Enterprise Data on the LAN threat
6. The Outsider Attacking Enterprise Data Outside the LAN threat
7. The Malicious Software threat

1.1 The following sections describe:

– why most Information Protection treatments fail to be scientific/systematic,
– why most Information Protection treatments fail to motivate the material,
– the nature of the threat model,
– how the model can be used to arrange Information Protection material in a scientific/systematic fashion,
– how the model facilitates a threat-first presentation of Information Protection techniques and
– why the classes of the threat model constitute all threats.

2. WHY CURRENT INFORMATION PROTECTION TREATMENTS FAIL TO BE SCIENTIFIC/SYSTEMATIC

A recent ad for a new information security book announces that the book covers the following major themes: (sub topics are also given for each major theme)

Cryptography, **Access Control**, **Protocols** and **Software**

A scientific minded reader is bound to ponder:

- Do these topics cover all facets of Information Protection?
- What are the relationships between these different themes?
- Is there a significance to the given order?

In general, Information Security or Information Protection books present topics in a somewhat ad hoc manner that does not lend itself to answering these questions. This, in turn, lessens their effectiveness as teaching tools. At a minimum every reader would like to know if an Information Protection book covers all aspects of information protection.

The ability to answer this question and the others posed above requires a systematic approach, which is a direct consequence of following the scientific method. The Merriam-Webster on-line dictionary[1] defines "scientific method" as:

> "**:** principles and procedures for the systematic pursuit of knowledge involving the recognition and formulation of a problem, ... "

The lack of a scientific approach is noteworthy, since, we are, after all, computer scientists. The periodic table of elements is a good example of a systematic arrangement of information that provides answers to these questions. It does list all currently known elements, it does show the relationships between different classes of elements (Alkali metals, Alkali earth metals, Transition metals, Rare earth metals, Noble gases and Halogens) and there is a significance to the order of elements in the table.

The result of this systematic arrangement of elements is a framework that facilitates an understanding of their properties.

The threat model presented here systematically arranges the various topics of Information Protection. Thus, it facilitates learning in the same way that the periodic table of elements facilitates learning.

3. WHY CURRENT TREATMENTS FAIL TO MOTIVATE THE MATERIAL

Information Protection and First Aid are similar in the sense that they are both problem/solution disciplines. If a First Aid course is organized around solutions the approach is less than optimal and fails to motivate the material. Such a presentation might start with aspirin and describe what aspirins do and what aspirins don't do. It might then present bandages and describe what they do and what they don't do. And so forth.

This material is better presented if it is organized around the problems. For acid indigestion do this and this and this. For bleeding do this and this and this. And so forth. When presented this way the student's interest is first peaked by the problem. "Hmm, how is acid indigestion addressed?" Then when the solution is presented it has context and meaning. The student is waiting to hear the solution and is likely to better comprehend it because of the context.

Currently most Information Protection books are arranged around solutions, like Cryptography, Access Control, Protocols, etc. And often the context is ignored. When execution domains are presented how often is the threat that they address also presented? If the student does not know what the problem is they are not going to be tuned into the solution. From a pedagogical standpoint a better approach is to arrange Information Protection topics around the problems and then present the protection techniques that address them.

Since the threat model introduced here arranges Information Protection material around the problems it naturally motivates an engaging presentation of protection techniques.

4. THE THREAT MODEL

The threat model presented here is based on a collection of seven threat classes that are mutually disjoint and together constitute all threats. These classes are based upon a typical enterprise computing situation, like the one shown below. This situation includes two Local Area Networks (LANs) that are physically far apart so they communicate via a Wide Area Network (WAN), like the Internet. The goal of the enterprise Information Protection policy is to protect enterprise information from unauthorized observation, modification and denial of availability and to protect the authenticity of source information for enterprise messages.

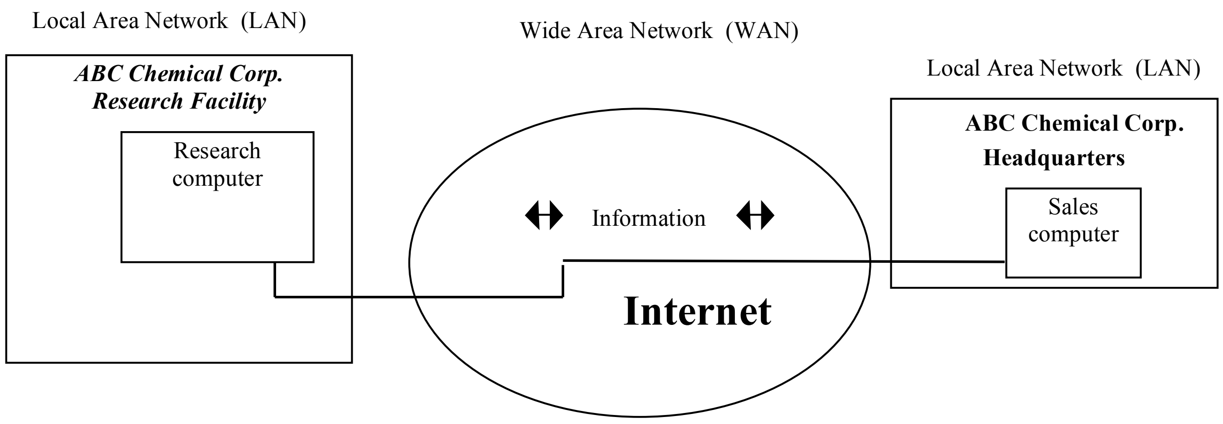


Figure 1. Typical Enterprise Computing Environment

Persons that are authorized to access enterprise information (such as employees) are called “insiders.” Persons that are not authorized to access enterprise information are called “outsiders.” The LAN systems and network infrastructure are assumed to be protected by typical LAN physical and personnel security measures.

This means that information residing on the LAN systems or on the LAN wired networks does not need the same type of protection that is required for information that is residing on systems and networks outside the LAN or in a wireless portion of the LAN.

The model refers to information residing on the LAN systems or on the LAN wired networks as “***information on the LAN***” and information that is residing on systems and networks outside the LAN or in a wireless portion of the LAN as “***information that is outside the LAN***.”

The seven threat classes of the model are:

(1) Non-Human threat
(2) Human Error threat
(3) Authorized Person threat
(4) Unauthorized Insider threat
(5) Outsider Attacking Enterprise Data on the LAN threat
(6) Outsider Attacking Enterprise Data Outside the LAN threat
(7) Malicious Software threat (orchestrated by participants of classes (4) or (5))

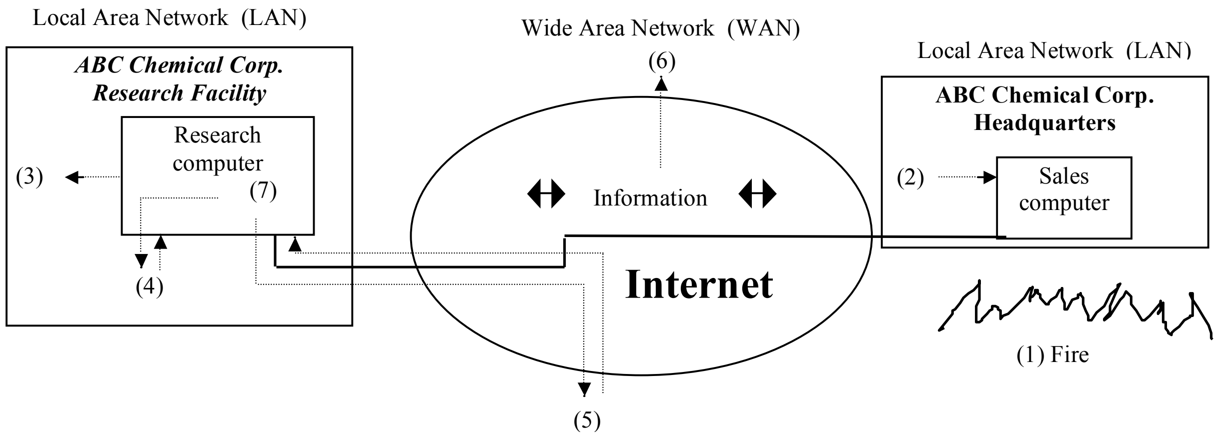


Figure 2. Graphical representation of the seven threat classes

Examples threats in each threat class are given below.

Table 1. Examples of each threat class

Threat class	Example
(1) Non-Human	Fire
(2) Human Error	An employee accidentally deletes a file
(3) Authorized Person	A researcher sells some research data to a competitor
(4) Unauthorized Insider	A mail clerk circumvents the file Access Control mechanisms and obtains some research data
(5) Outsider Attacking Enterprise Data on the LAN	Someone in Iceland breaks into the research computer and obtains research data
(6) Outsider Attacking Enterprise Data Outside the LAN	Someone in Florida intercepts microwave traffic and captures enterprise data that is flowing across the Internet
(7) Malicious Software	Someone in Iowa plants a Trojan horse in a music sharing program

5. HOW THE MODEL SUPPORTS A SCIENTIFIC/SYSTEMATIC TREATMENT OF INFORMATION PROTECTION MATERIAL

The following table lists the threat classes and general categories of protection techniques that address each threat class. Viewed this way it is possible to see how the categories of common Information Protection techniques (Passwords, ACLs, Firewalls, Cryptography, etc.) fit into the big picture of Information Protection. And since these threat classes contain all threats (see section 7 for the proof) complete protection against all threats is achieved if each threat class is adequately addressed.

Table 2. Categories of protection techniques that address each threat class

Threat Class	General Protection Measures
Non-Human (1)	**Physical Security**
Human Error (2)	**Error Security** Training Secure Defaults Simple interfaces
Authorized Person (3)	**Personnel Security** Personnel Security (background checks) Separation of Duties Audit
Unauthorized Insider (4)	**Operating System Security** Identification and Authentication File <u>Access Control</u> Memory Protection (Process Separation, Rings)
Outsider Attacking Enterprise Data on the LAN (5)	**Perimeter Security** Firewalls (Network Traffic <u>Access Control</u>)
Outsider Attacking Enterprise Data Outside the LAN (6)	**Communications Security** Cryptographic <u>Access Control</u>
Malicious Software (7)	**Malicious Software Security** Trojan horses: MAC policies Worms: Assurance techniques Viruses: Ad Hoc Antivirus techniques

Although the assignment of these general categories of protection techniques to the seven threat classes is somewhat consistent it is not perfect. For example, parity bits and backup systems, which are used to address Non-Human threats, are not Physical Security techniques. Similarly, the cryptographic technique of Digital Signatures (which is part of Communications Security) is needed to protect the authenticity of message source information against threat classes 4 and 5. And particularly noteworthy is the omission of Physical Security as a technique for addressing the Unauthorized Insider. If an Insider can get physical access to a server system they can access all its data in spite of Operating System Security (I&A, ACLs and Memory Protection). They simply steal the hard drive or boot the server off of removable media that contains hacking tools.

These exceptions do not detract from the threat model, though. They simply mean that the names of the general categories of protection techniques (Physical Security, Error Security, Personnel Security, etc.) that address the classes of the threat model are not perfect. The names listed above were selected because most of them are common terms (Physical Security, Personnel Security, Perimeter Security, Communications Security) and they are sufficiently correct since they encompass most of techniques that are used to address the threat classes.

Since many Information Protection measures are based on Access Control there are a few supplementary Information Protection techniques that specifically address failures in Access Control mechanisms. For completeness the model needs an additional section that contains these techniques.

5.1 Techniques that address failures in Access Control mechanisms

The table below identifies three techniques that are frequently used to address failures in Access Control mechanisms.

Table 3. Common Techniques for Addressing Access Control Failures

Access Control Faliures	• Design and Development Principles that Promote a Sufficient Degree of Assurance • Defense in Depth • Intrusion Detection

The technique of system Assurance is presented first. When a Web Server program has a Buffer Overflow vulnerability that lets an unauthorized user access the system it is a failure in an Access Control mechanism. This may seem odd because we generally do not consider a Web Server program as part of a system's Access Control mechanisms. It is, however. Implicitly we expect the Web Server program to not grant system access to unauthorized users. This is typically not an explicit policy but it is an implicit policy. When such a program does grant an unauthorized user system access it is failing to enforce this implicit policy and hence it is an instance of a failure in an Access Control mechanism. Thus, design and development techniques that promote assurance or trustworthiness in systems address this type of problem.

Defense in Depth is another technique that can effectively address failures in Access Control Mechanisms. In a Defense in Depth strategy more than one protection mechanism is used in series to protect an asset. Two Access Control mechanisms are in series if access to an asset requires the permission of both mechanisms.

A Defense in Depth strategy is optimal when failures of the individual the Access Control mechanisms are statistically independent events. When this is the case the likelihood of concurrent failures in all Access Control mechanisms is the product of the failure rates of each individual mechanism. Typically this greatly reduces the overall failure rate.

Intrusion Detection is another technique that is commonly used to address the problem of Access Control failures. In general, Intrusion Detection techniques look for activities that are either a result of an Access Control failure or an unauthorized action, such as port probing, that is not easily prevented by Access Control techniques.

6. HOW THE MODEL SUPPORTS A THREAT-FIRST PRESENTATION OF INFORMATION PROTECTION TECHNIQUES

A straightforward application of the threat model to the presentation of Information Protection material could simply allocate a section or chapter to each threat class, such as is done below.

Table 4. Applying the Threat Model to the Presentation of Information Protection Material

Chapters	General Information Protection Measures
1. Non-Human Threat	**Physical Security**
2. Human Error Threat	**Error Security**
3. Authorized Person Threat	**Personnel Security**
4. Unauthorized Insider Threat	**Operating System Security**
5. Outsider Attacking Enterprise Data on the LAN Threat	**Perimeter Security**
6. Outsider Attacking Enterprise Data Outside the LAN Threat	**Communications Security**
7. Malicious Software Threat	**Malicious Software**
8. Access Control Failures	**Access Control Failure Techniques**

In order to instill interest in the student the presentation of each protection technique should begin with the threat that is addressed by the protection technique. Chapter 4 above demonstrates this well.

Hypothesize a system with no user Identification capabilities and show how an Insider user can access information that they are not authorized to access.

Hypothesize a system with a user Identification capability and a file access control mechanism (e.g., ACLs and ACL program) but no user Authentication capability and show how an Insider can access information that they are not authorized to access.

Hypothesize a system with user Identification and Authentication and a file access control mechanism (e.g., ACLs and ACL program) and

- consider storing the password information in the clear,
- consider storing the password information encrypted with conventional cryptography,
- consider storing the password information in a hashed format.
  - Introduce the Dictionary Attack.
  - Introduce the Brute Force Attack.
  - Go over password selection guidelines

Hypothesize a system with user Identification and Authentication (I&A) and a file access control mechanism (e.g., ACLs and ACL program) but no process address space control and show how an Insider can directly read and write bytes of memory and access information that they are not authorized to access.

Hypothesize a system with user Identification and Authentication, a file access control mechanism (e.g., ACLs and ACL program) and process address space control but no ring mechanism and show how an Insider can bypass the file access controls and ultimately access information that they are not authorized to access.

The conclusion of Chapter 4 is that all aspects of OS Security are necessary to adequately address the Insider threat. As previously mentioned, Physical Security also needs to be mentioned because it too plays a role in addressing the Insider Threat.

This arrangement of material is starkly different from what is commonly done today. Most treatments today have a chapter on I&A, a chapter on file access control policies and mechanisms and a chapter on Operating System Security that presents rings. The arrangement above shows that all these techniques work in concert together to address the same threat, which is the Unauthorized Insider threat.

Chapter 7 (Malicious Software Threat) above also demonstrates the value of this approach. Since Mandatory Access Control (MAC) policies are primarily implemented to address Trojan horse programs in applications it only makes sense to present the threat (Trojan horses) and the protection technique (MAC policies) in the same chapter.

Traditional treatments obscure the connection between MAC policies and Trojan horse programs by presenting MAC policies (along with Discretionary Access Control policies) in a Policies chapter and Trojan horse programs in a Malicious Software chapter. When presented this way the reader has to make the connection between the purpose of a MAC policy and the Trojan horse program threat on their own.

7. WHY THE SEVEN CLASSES INCLUDE ALL THREATS

The follow figures prove that the seven classes of the model do, in fact, include all threats. The proof involves repeatedly partitioning the set of all threats until the 7 classes of the model are derived. First, the set of All Threats is partitioned into **Human-Based threats** and **Non-Human-Based threats**.

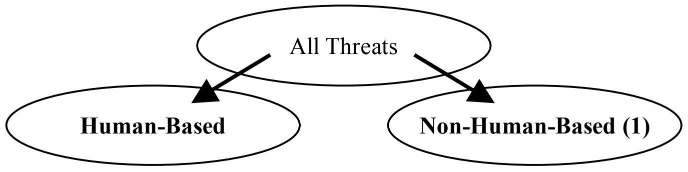


Figure 3. Partitioning all threats into Human and Non-Human-Based threats

Then the set of Human-Based threats is partitioned into the set of **Intentional threats** and **Accidental threats**. Intentional threats are, by definition, performed by Untrustworthy Persons. With the exception of Non-Human threats (fires, lightening, etc.) and Accidental Errors all threats are a result of Untrustworthy Persons.

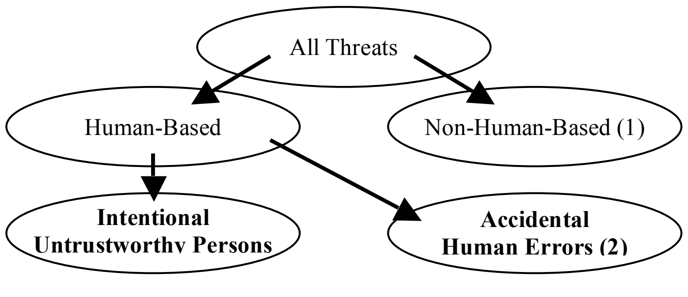

All Threats
Human-Based
Non-Human-Based (1)
Intentional Untrustworthy Persons
Accidental Human Errors (2)

Figure 4. Partitioning Human-Based Threats into Intentional and Accidental Threats

Untrustworthy Persons come in two varieties, those that are authorized to access the data and those that are not authorized to access the data. Thus, the set of Untrustworthy Persons is partitioned into **Unauthorized Persons** and **Authorized Persons**.

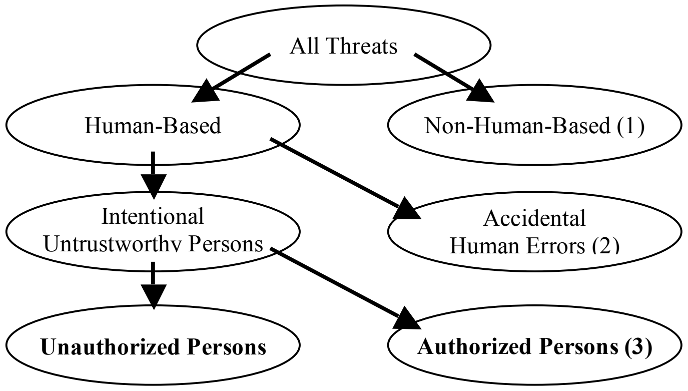


Figure 5. Partitioning Untrustworthy persons into Unauthorized and Authorized sets

Unauthorized Persons come in two varieties, those that are Insiders and those that are Outsiders. Thus, Unauthorized Persons are partitioned accordingly.

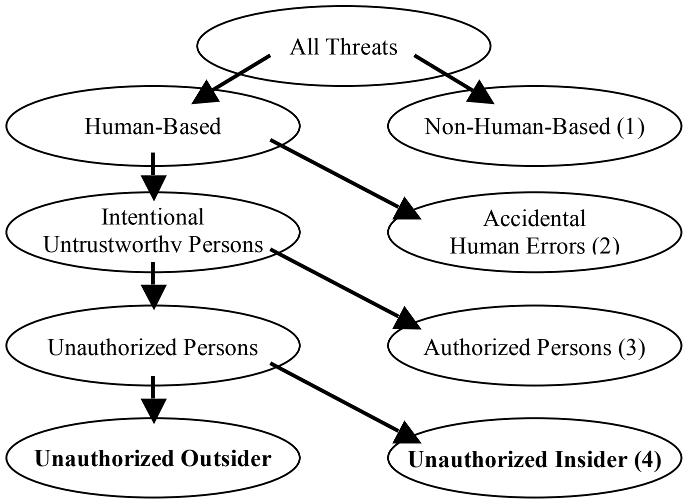


Figure 6. Partitioning Unauthorized Persons into Outsiders and Insiders

Outsiders can threaten enterprise data in two different places, in the environment outside the LAN or on the LAN. Thus, Outsiders are partitioned accordingly.

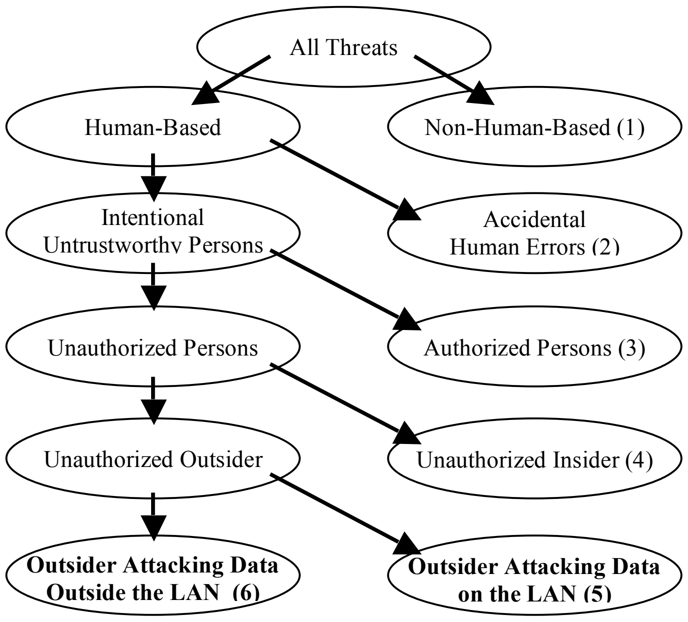

All Threats
Human-Based
Non-Human-Based (1)
Intentional Untrustworthy Persons
Accidental Human Errors (2)
Unauthorized Persons
Authorized Persons (3)
Unauthorized Outsider
Unauthorized Insider (4)
Outsider Attacking Data Outside the LAN (6)
Outsider Attacking Data on the LAN (5)

Figure 7. Partitioning Outsiders into LAN Data Attackers and Non-LAN Data Attackers

A seventh threat class (**Malicious Software**) is introduced because Unauthorized Insiders and Outsiders Attacking Enterprise Data on the LAN can attack LAN information in two ways, directly and indirectly. Direct attacks are prevented by appropriate Access Control techniques. This is due to the nature of Access Control, it grants access to authorized persons and denies access to unauthorized persons and both the Unauthorized Insiders and the Outsiders Attacking Data on the LAN are not authorized for the targeted data.

Indirect attacks are generally not addressed by typical Access Control techniques. Indirect attacks use Malicious Software to attack "from the inside." This type of attack is so fundamentally different it is broken out into its own threat class. Access Control techniques keep unauthorized users from gaining direct access, they do not prevent a victim from unknowingly sending their files to a website in Eastern Europe when they execute a Trojan horse program. If the victim is authorized to send data to a website Access Control cannot address this problem.

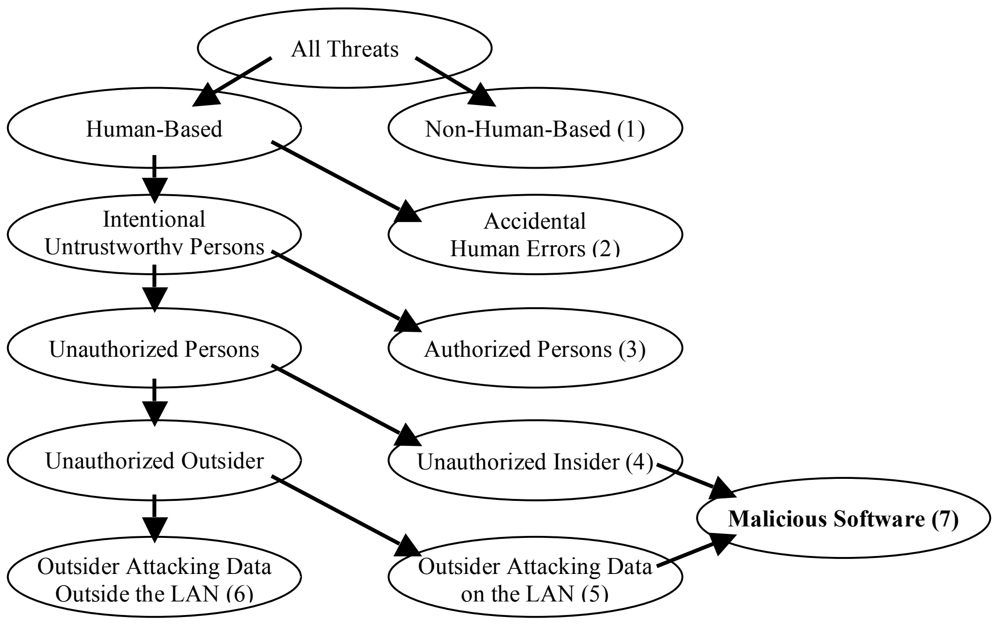


Figure 8. Introducing the Malicious Software Threat Class

Since the seven classes above are derived by partitioning the set of all threats they must contain every information threat. As such, adequately addressing each threat class provides complete coverage against all threats.

8. CONCLUSION

The given threat model provides a systematic way for presenting Information Protection topics. Systematic presentations, like the one proposed here, enhance learning by showing how all the pieces fit together and by establishing a sense of complete coverage against all threats. The use of the threat model presented here also promotes learning because it organizes the material around the threats that provides context and necessity for each protection technique.

ACKNOWLEDGEMENTS

I would like to thank Thuy Nguyen for her valuable comments.

REFERENCES

1. The Merriam-Webster on-line dictionary is at www.m-w.com.

NETWORK SECURITY AND INFORMATION ASSURANCE AT JACKSON STATE UNIVERSITY
A Successful Story

Houssain Kettani

Department of Computer Science, Jackson State University, Jackson, Mississippi, 39217

Abstract: In this paper we share the story of Network Security and Information Assurance education at Jackson State University. We narrate its start, collaboration with another university, and our steps towards establishing a Center of Academic Excellence in this domain.

Key words: Network Security, Information Assurance, Education, Jackson State University

1. INTRODUCTION

The example of the Internet and communication networks in the progress of our lives nowadays is water. We cannot imagine our economy, education, communication with others, defense, pleasure, etc, without the Internet. E-mail, HTTP, WWW, all are thought nowadays to be necessities, although we know for sure that Adam and Eve did not have them! Consequently, the protection of information becomes more and more challenging. Accordingly, the interest in Network Security and Information Assurance (NSIA) is growing day by day and the latter is posed to be in the heart of Computer Science education.

Acknowledging this fact, the Department of Computer Science at Jackson State University (JSU) moved into providing NSIA as part of its curriculum. Initially, a network security course for undergraduates and graduate students was offered once a year in collaboration with Mississippi State University (MSU), Mississippi State, Mississippi. Thus, in spring 2002, this course was offered as a distance learning class. Eventually, the course was offered in spring 2004 by Dr. H. Kettani, a faculty member of the Department of Computer Science at JSU and is being offered in every spring semester

Meanwhile, the department has received two generous grants from the National Science Foundation to enhance the department's NSIA program and establish a Center of Academic Excellence in NSIA. As a result, two undergraduate and two graduate students have been supported and are performing very well in their NSIA career.

In this paper, we elaborate on the journey of the Department of Computer Science at JSU to establish NSIA awareness in its program, our success, and future plans.

2. ABOUT JACKSON STATE UNIVERSITY

Jackson State University (JSU) is one of the nation's largest and prominent Historically Black College/University (HBCU) located at Jackson, the capital of the state of Mississippi. As of fall 2003, Jackson State University, enrolled 7,815 students. Of this total enrollment, 63% are female, 81% are undergraduates, and 94% are African Americans [1].

The College of Science, Engineering, and Technology at Jackson State University provides an excellent opportunity for large numbers of the African Americans, both men and women, to obtain engineering education. As of fall 2003, this college enrolled 1,655 students, which is 23% of the total enrollment of the university. Of this college's total enrollment, 52% are female, 87% are undergraduates, and 94% are African Americans [1].

The Department of Computer Science offers both Bachelor of Science and Master of Science degrees. The Bachelor of Science degree program is accredited by the Accreditation Board for Engineering and Technology (ABET). The Department of Computer Science is also a leading producer of African Americans with the Bachelor of Science degree in Computer Science. As of fall 2003, this department enrolled 330 students, which is 20% of the total enrollment of the College of Science, Engineering, and Technology. Of this department's total enrollment, 38% are female, 76% are undergraduates, and 84% are African Americans [1].

3. COLLABORATION WITH MISSISSIPPI STATE UNIVERSITY

Mississippi State University (MSU) has developed a strong NSIA program and is two hours drive north east of JSU. Thus, when we were thinking of developing our NSIA program, we decided that it would be a good idea to learn from MSU’s experience and collaborate with its Computer Science Department. Accordingly, a network security course for undergraduates and graduate students was offered once a year in collaboration with MSU as distance learning class. Thus, in spring 2002, the course was offered at MSU by Dr. R. Vaughn, a faculty member of the Department of Computer Science at MSU and was broadcast to JSU. In spring 2004, the course was offered as semi distance learning class, where Dr. Vaughn alternated between MSU and JSU. In addition, this course had a laboratory that was offered at JSU and supervised by two JSU graduate students as teaching assistants. The adopted lab experiments followed the lab manual that was developed at MSU.

4. NATIONAL SCIENCE FOUNDATION’S SUPPORT

In response to a National Science Foundation (NSF) request for proposals for a Scholarship for Service (SFS) grant, and in collaboration with MSU, JSU was awarded in August 2002 two SFS grants of $350,000 for a period of 4 years. The purpose of these grants is to increase NSIA awareness on campus. With these grants we were able to support two undergraduate students and two graduate students for duration of two years. These students were required to have an internship in NSIA domain at a government institution of their choice, and work for two consecutive years at such institution after the completion of their degree. We note that the students that we supported did not have trouble in finding such internships. In the past two summers, we had one graduate and one undergraduate student who worked at the Federal Bureau of Investigation (FBI) at Washington, DC, a graduate student who worked at the Federal Administration Agency (FAA) at Washington, DC, and an undergraduate student NASA Stennis Space Center, Mississippi. In May 2005, our first supported student has earned her Masters of Science and started working at another Government agency.

The grant also helped in sending students and faculty members to attend educational and training conferences and workshops in NSIA domain. For example, in 2004, Dr. Kettani was sent to two conferences organized by SysAdmin, Audit, Network, and Security (SANS) Institute. Dr. Kettani also attended the Sixth Workshop on Education in Computer Security (WECS6). Thus, in spring 2005, he instructed the course independently from MSU. The course still has a lab, and the enrollment is a dozen to a score of students.

5. TOWARDS A CENTER OF ACADEMIC EXCELLENCE

The first step in establishing a Center of Academic Excellence in NSAI is to map the courses that we offer in NSIA domain to two standards of the Information Assurance Courseware Evaluation (IACE). Accordingly, in February 2004, the Review Committee of IACE has validated the mapping at 100% for the Committee on National Security Systems (CNSS) National Standards 4011 and 4014 Entry Level. Consequently, JSU has received recognition and a certificate during the CNSS Annual Conference, Norfolk, Virginia, April, 2004. Consequently, we have applied to the 8th annual NSA and DHS for the Centers of Academic Excellence in Information Assurance Education Program Offering in December 2005.

ACKNOWLEDGEMENTS

This work was supported in part by the National Science foundation (NSF) and the Naval Postgraduate School, Monterey, California. The authors are thankful to Dr. R. Vaughn from Mississippi State University (MSU), Mississippi State, Mississippi, for his collaboration and dedication to initiate our NSIA program.

REFERENCES

1. Jackson State University Fifteen Year Report of Student Enrollment by Academic Disciplines, Classification and Gender 1988-1989 to 2003-2004, Office of Institutional Research and Planning, November, 2003.

TWO SUCCESSFUL MINIPROJECTS IN AN OVERVIEW INFORMATION ASSURANCE COURSE

Judith L. Gersting

University of Hawaii at Hilo

Key words: Steganography, Information Assurance Plan, Information Assurance Policy

1. INTRODUCTION

In the Computer Science Department at the University of Hawaii at Hilo, we have recently introduced an undergraduate Information Assurance course. This is a senior-level elective course for computer science majors, and is intended to provide a general overview of the field.

Information assurance can be somewhat overwhelming to teach because its scope is so broad. It touches on many areas of computer science (programming, operating systems, database, networks), as well as mathematics (cryptology), and management, legal, and ethical issues. But it is a very important topic, one in which there is a great deal of student interest and for which there is no end of related items to be found in the news.

Prior to attending the WECS6 workshop I had taught a seminar on cryptography and network security, and, later, the first version of our overview course. I found materials for "talking about" lots of things. But, for our students at least, talking often goes right over their heads and nothing much sinks in until they engage in some concrete learning experience. So I was looking for additional topics that I could translate into student assignments or lab exercises, and, based on information from WECS6, I had the germ of two ideas. I wanted to do something with steganography, and something with security planning. These formed the bases for two sets of student activities/assignments that were incorporated in my second version of our overview course, taught in Spring 2005.

These activities proved relatively successful based on my criteria for success, which were:

a. Students learned something that was new to them.
b. A "real world" security problem or issue was explored.
c. Students found the activities enjoyable.

2. STEGANOGRAPHY

I was intrigued by the demonstration on steganography at WECS6. Steganography is the art of hiding secret information in plain sight, and in various forms, it dates back to ancient times. Today it generally involves highjacking the least significant bits of an image file to store secret information such as a text file, and with the ready transmission of images over the Web, it is a potential tactic for instantaneously broadcasting secret information in plain sight around the world.

The assignment package I put together on this topic involved the following student activities:

a. Students reported on historical uses of steganography.

b. Students used a free steganography software tool called S-Tools [6], which allows for least significant bit encryption of data within an image file. In a closed lab exercise, students explored this tool, answered several questions about how the tool works, and investigated four copies of a 900 KB image file I had prepared. Each looked the same; one was nothing but the image and the other three held increasingly larger files. Students were impressed with the fact that a 650 KB text file - an entire book chapter - could be embedded in this image and appear quite undetectable.

c. I provided information on the BMP file format as background for a programming assignment that consisted of three parts. Students worked in small teams to write (in C++) three programs that constitute a simple version of S-Tools to meet the following specifications:

i. As a first pass, read in and write out a BMP image file. The output image should, of course, look like the input image.

ii. Read in an image file and a text file and encode the text file sequentially in the least significant bits of the image file.

iii. Read in a steganographic image file containing text encoded as in part ii and recreate the original text file.

d. In a closed lab session, each team used a BMP image file and a text file of their choosing, encoded the text file in the image file, and passed the steganographic image file to another team, who attempted to decode it and recover the original text file. This proved a bit more difficult than anticipated because, while each team had tested their programs on their own files, some of the programs (either the encoding or the decoding) didn't work correctly when paired with another team's program. To pin down the errors, the teams began trading text and image files and by the end of the hour one team still had an encoding error (later fixed) but all other code was working properly.

e. One student took this project a little farther. Using two image files, one quite complex and one rather simple, he inserted random bits into each image, first randomizing just the low order bit, then the two low order bits, 3 bits, 4 bits, up to 7 of the 8 bits, and wrote out the image file for each case. Two interesting observations could be made. One was that even with 7 out of every 8 bits mangled, the original images were still discernable. The other was that, contrary to what one might expect, the more complex image absorbed the mangled bits less successfully than the simple image, that is, slight variations or degradations appeared visible in the complex image with fewer mangled bits than in the simple image. This little experiment will be included as part of the overall steganography assignment package when the course is next taught.

3. SECURITY PLANNING

The second assignment package, which came near the end of the semester, involved management issues, specifically security planning. A security plan for an organization should address the following:

- Risk analysis
- Security policy
- Who is accountable for what
- Timeline for change
- Structure for periodic revision

Risk analysis involves making an inventory of the organization's assets, determining how each might be vulnerable, how each might be protected, and the cost/benefit ratio of implementing such protection. The security policy, based on the risk analysis, spells out what the organization's security decisions are, what assets are accessible to whom and for what purpose, and what protections are in place. The security plan will also include assignment of responsibilities for security measures, a timeline for implementing new security controls, and a mechanism to ensure periodic review and revision of this whole process.

Students may have little opportunity to examine security plans, yet in the business world, security planning may indeed be something they are called upon to do. The assignment, partially based on exercises found in Chapter 8 of [5], asks students to develop security plans for fictional enterprises making use of the guidelines in the OCTAVE®-S Implementation Guide, Version 1.0, Volume 10 [3]. The OCTAVE-S Implementation Guide can be downloaded from http://www.sei.cmu.edu/publications/documents/04.reports/04hb003.html

3.1 The Octave-S Guidelines

The OCTAVE method [1] was developed at the Software Engineering Institute at Carnegie Mellon University. It involves specific steps for building a security plan for large organizations, i.e., those with over 300 employees. (More information on the OCTAVE method can be found in [2].) OCTAVE-S was developed later as a more streamlined version suitable for small organizations.

The OCTAVE-S Implementation Guide is a 10-volume series that includes documentation and guidance for use of the OCTAVE-S system. Obviously a complete application of the OCTAVE-S system would be a major effort, and training is recommended. Volume 10 of the series, however, is an example scenario of the OCTAVE-S methodology applied to a fictitious medical facility called MedSite. This is the document made available to students for this assignment.

Most valuable in this example scenario are the worksheets illustrating asset identification, impact evaluation, identification and evaluation of current security practices, identification of critical assets and their risk profiles, protection strategies, recommendations, and action plans.

3.2 The Assignment

Part of the assignment statement given to students follows:

"Your job is to write a security plan for a fictional enterprise. Obviously, the Octave-S example in Volume 10 is more than is required for this assignment, but you should read it to gain some ideas for making lists and/or tables about your fictional enterprise. Your first task, of course, is to fill in some details of your fictional enterprise so that you can then identify its most important assets, determine in what ways and how likely they are to be vulnerable, estimate the costs of that vulnerability, know who is responsible for these assets, and so forth. Your report should be on the order of 12-15 pages, and should be an example of professional writing."

This was also a team project. Each team had a different fictional enterprise, namely

- A credit union
- A regional airline
- An electric utility company
- A political campaign headquarters

Students had less than two weeks to complete the project.

3.3 The Results

This was a very successful assignment in terms of student enjoyment, although they all wished they had had more time (and in this case this familiar student lament may have been justified). The fictional nature of their enterprises gave the students the freedom to "invent" assets, while the worksheet formats from the OCTAVE-S Volume 10 document helped to provide structure for their reports. All team papers were well-written, and used selected worksheet formats from the OCTAVE-S Volume 10 document.

The political campaign team had the greatest difficulty identifying assets. The credit union team thought that their enterprise was rather dull. The most interesting report came from the electric utility company, which early on decided to be a hydroelectric facility, and then found an actual risk assessment methodology document applicable to their enterprise [4].

By the time this assignment was completed, a graduating student in this course had his job offer in hand and had learned that one of his responsibilities was going to be to write a security plan for the organization hiring him. This certainly added interest in this assignment for him, and for his fellow students.

4. CONCLUSION

My objective after attending WECS6 was to again offer the overview security course, but to incorporate more hands-on activities or assignments than previously. Using material inspired from WECS6, the unit on steganography took about 3 hours of class and closed lab time. The unit on management issues took about 4 hours of class time, including time for presentation of team reports. Other material from WECS6 was filtered throughout some of the other course topics. Nothing specific was done for or with underrepresented groups in computer science, although the University of Hawaii at Hilo is considered a minority institution. No publications resulted from this work, but new materials for assignments and activities were developed.

The IA course will be a part of our regular course rotation and will be offered every two years. We hope to continue to develop activities and assignments in various areas to further enrich student learning opportunities.

REFERENCES

1. Alberts, C., Behrens, S., Pethia, R., Wilson, W., Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) Framework, Version 1.0**,** Software Engineering Institute Technical Report, CMU/SEI-99-TR-017, 1999. http://www.sei.cmu.edu/publications/documents/99.reports/99tr017/99tr017abstract.html
2. Alberts, C., and Dorofee, A., Managing Information Security Risks: The OCTAVE (SM) Approach, Addison Wesley Professional, 2002.
3. Alberts, C., Dorofee, A., Stevens, J., Woody, C., OCTAVE$^{®}$-S Implementation Guide, Version 1.0, Volume 10: Example Scenario, CMU/SEI-2004-HB-003, 2005. http://www.sei.cmu.edu/publications/documents/04.reports/04hb003.html
4. Matalucci, R., Risk Assessment Methodology for Dams, Proceedings of the 6th International Conference on Probabilistic Safety Assessment and Management (PSAM6), 23-28 June 2002, San Juan, Puerto Rico, USA, Vol. I, pp 169-176. http://www.esisac.com/publicdocs/Workshop_Orlando/RAM-D_RM.pdf
5. Pfleeger, Charles P., and Pfleeger, Shari L., Security in Computing, Third Edition, Prentice Hall, 2003.
6. S-Tools 4.0, available at http://www.spychecker.com/program/stools.html

WECS6 FOLLOW-UP REPORT

Charles Anderson

Western Oregon University

Abstract: This report is a summary of my educational efforts in the field of computer and network security following my attendance at the WEC6 workshop and tutorials. In other words, what I did after my summer vacation in Monterey.

Key words: WECS6, Computer Science Education, curriculum, development, infosec.

1. OBJECTIVES

My initial objectives immediately following WECS6 were very modest: to enhance the coverage of security topics for those classes in which I typically discuss security – e.g., networking. However, during the course of the 2004-05 school year my objectives increased fairly dramatically. I decided to offer an introductory class in security as an unpaid overload, and I began to discuss the possibility of expanding our Computer Science (CS) major to include an emphasis in security. In preparation for that, I created a new block of course numbers (460 – 469) for security classes, and wrote course descriptions for two initial offerings.

In 2005-06, I am expanding on the curriculum development to complete the specification of the new security emphasis. I will be submitting two to four additional course descriptions for the new emphasis. I am further revising my existing courses with security as a primary learning outcome – i.e., when considering topics or presentations, one of my metrics is now "does this provide students with knowledge they need to understand security later in their education or employment?"

2. COURSES TAUGHT

In the 2004-05 academic year, I incorporated security or expanded the coverage of it in a number of classes that I usually teach. I also taught a new introduction to security course. Furthermore, as a direct result of the WECS6 conference, I added security to a new class, which initially seemed to have little to do with security.

2.1 Existing Courses

In this section, I discuss courses that I typically teach and how I've changed my presentation of security topics.

2.1.1 CS 350 – Network Administration – 3 units

This is a first course in networking aimed primarily at our Information Systems (IS) majors (non-programmers). It is focused primarily on LAN networking. From the WECS tutorial, I got some new ideas for presenting the networking topics (independent of security) from J. D. Fulp's introductory talk – e.g., using a spreadsheet to construct a template to overlay on binary data to show the idea of fields within a packet. This class includes a section on host-based and LAN-oriented security which was reorganized and greatly clarified based on information from the entire workshop. While teaching this class I had the epiphany that preparing students to understand network attacks should be a primary learning outcome of the course.

2.1.2 IS 452 – Internet – 3 units

This is the second class in networking, which is focused primarily beyond the LAN. It is a follow up to CS 350. In addition to general reorganization and clarification of security topcis, I applied information and materials that I developed for my Introduction to Information Assurance course (see below) to discuss topics such as VPNs and firewalls.

2.2 New Courses

2.2.1 CS 459 – Introduction to Information Assurance – 3 units

This was a brand-new course that I taught as a direct result of the WECS workshop. Because it was new, it was an elective for the students. Despite the fact that the students were not required to take the course, it was one of the largest/most popular courses I've ever taught. It was available to our more senior CS and IS students – i.e., programmers and non-programmers. On the first day after dealing with the typical administrative chores, I showed the *Strategic Cyber Defense* DVD that we got at the tutorial; the students got a kick out of the video.

I used *Corporate Computer and Network Security* by Panko as the textbook. We covered chapters one through eight. Topics included access control, physical security, attack methods, firewalls, host security, and some basic cryptography. I don't recall this book being mentioned at the workshop, but I found it very useful because it provides a decent introduction to a broad range of topics, and it doesn't dive into cryptography immediately or in too much depth, which would be rather challenging for our students. It has some rough spots that I hope will be addressed in the next edition.

2.2.2 CS 272 – Low Level Programming – 3 units

This class covers programming in C and assembly language. It is a rarely taught elective, and this was the first time I taught it. This is a class in which I initially had no intention of discussing security. However, as it turned out, the majority of the assignments were security-related. Based on a talk by Everett Bull at the WECS conference, I got a copy of the textbook *Computer Systems: A Programmers Perspective* by Bryant and O'Halloran and gave the students two programming assignments from the book. The first was the "bomb defusing" exercise that teaches the students how to use a debugger and work with disassembled binaries – i.e., reverse engineering. The second assignment involved creating stack overflows in a program. In addition, as an exercise in algorithm design and optimization, I had the students write a brute-force password cracking program, and we compared the speeds of their implementations. The idea to do this was a result of a talk at the tutorial by one of the CISR faculty (whose name I have forgotten) discussing password strength and cracking times.

3. CIRRICULUM DEVELOPMENT

At WOU, Computer Science students select an emphasis or elective sequence as part of their major. An emphasis is composed of three senior-level courses in a given subject area. Based on my experiences at the WECS workshop, I began discussing the idea of creating a new emphasis in computer and network security. I did not have enough time to completely flesh out the details of the new emphasis during the 2004-05 school year, but I did create course descriptions for two new courses. The courses have been approved by the campus curriculum committee and will appear in next year's catalog.

During the 2005-06 year, I will be completing the definition of the security emphasis, and I will be submitting course descriptions for the new courses. I am hopeful that these will be approved this year.

Although these courses are nominally intended to be part of our Computer Science major, it is my intention to make as many of them available to our Information Systems majors as well. Both majors have separate networking and operating systems courses, but the new courses accept prerequisites from either major. At this time, the only course/topic I see as exclusively for CS majors will secure programming because students majoring in IS often choose the major to minimize the programming that they must do.

3.1 New Courses

Introduction to Information Assurance - CS 460: this is basic survey class. It will be an "official" version of the CS 459 class I taught last year and will cover a similar selection of topics. (The CS 459 course number I used before was just a temporary kludge to be able to teach the course without waiting for formal approval of an official course number.) I hope to make this a required course for all CS and IS majors, not just the CS majors who select the security emphasis.

Special Topics - CS 469: this course was added immediately to allow flexibility to offer classes in more specialized topics before official course numbers are available. Once such course numbers are available, this course may be used for unique topics or seminars.

The previous two courses have already been approved. Our department hasn't finalized the other courses in the security emphasis, but I am considering the following possibilities. Network Attack and Defense: a discussion of tools and techniques for attacking systems and protecting them from such attacks. Computer Forensics: tools and techniques for forensic analysis of computer systems. Secure Programming: an introduction into secure programming practices – e.g., avoiding buffer overflows, SQL injection attacks, etc.

4. OTHER NOTES

In this section, I discuss various little topics related to WECS and my teaching endeavors.

At the conclusion of my Introduction to Information Assurance class, I discussed the Scholarship For Service (SFS) program, and I shared information about the graduate program at the Naval Postgraduate School. One of my students, Justin Hoeckle, was very interested in SFS. He was accepted to a number of schools, and is now studying at Johns Hopkins. (He didn't apply to NPS because of desire/need to be on the east coast.)

With regards to under-represented student groups, our university attracts a large number of students who are the first in their family to attend college. Justin Hoeckle was one of these, and his success is due, in small part, to the security class he took from me. Although the state of Oregon doesn't recognize Asians as minorities, they are quite rare on our campus, and one of them, Minh Nguyen, took all of my networking classes and the security class. He was accepted to Cornell for graduate school but was unable to attend due to a lack of financial aid. (As a citizen of Vietnam, SFS was not an option for him.)

I have not developed any significant new materials related to security, other than the usual assignments and PowerPoint presentations. I am considering developing some readings for my networking classes because I cannot find textbooks that cover the topics that I want to cover. Such materials would have preparation for studying security as a major learning objective. I have not published anything related to my participation in WECS.

After "dipping my toe in the water" at WECS, I have immersed myself in reading computer security books, both for self-education, as well as looking for textbooks for future classes. In the course of this, I realized that computer forensics can be thought of as an in-depth study of operating systems and how they represent and store data. Based on that observation, I plan on teaching my advanced operating systems class this spring centered around forensics.

This interest in forensics has already paid off: shortly before Thanksgiving 2005, I got a phone call from Detective William Wiltse of the Salem, Oregon police department. He is one of two detectives on the force working in forensics. He is looking to develop a new tool for recovering fragments of MPEG movies. I hope to involve some students in the project, or failing that, I will work directly with him.

Even before his call, I had been considering how to join forces with our campus Criminal Justice department and possibly the police academy that shares our campus. At the very least, I would like to incorporate some information about the legal/law enforcement side of forensics for use in my class. A longer term project is to create a computer forensics course available to the Criminal Justice majors who lack the computer and networking background of our CS and IS majors.

One issue that was touched on at WECS that I am still working on is the "ethics" of teaching classes that involve potentially dangerous tools or techniques (e.g., hacking). To date, I have written a section in my course policy statements about "dual use technologies," but I still feel compelled to downplay my discussion or use of such tools. In order to be more effective, I need to feel comfortable with these topics and tools. I am hopeful that once our computer security emphasis is in place, I can have a rational discussion with campus-level administrators to get more official approval of such teaching.

Another positive item I took away from WECS was learning about using virtual machines like VMWare for security work. Prior to WECS, I knew of VMs, but didn't think much of them. I saw a number of speakers talk about and use VMWare for security research and demonstrations. When I received an offer from VMWare for a free academic copy of VMWare, I jumped on it. I am investigating how to get low-cost access to VMWare for an entire lab of computers in the CS department. In the meantime during my Introduction to Information Assurance course, a number of my students used Microsoft's Virtual PC (which we have free access to) for studying spyware. They were very impressed with the capabilities of VMs; Virtual PC made their project work much quicker and easier.

One final thing I learned about from a presentation at WECS ("Teaching Computer Security to Undergraduates" by Tikekar) is that our sister institution, Southern Oregon University, has started a degree program in computer security and information assurance. It is ironic that I had to travel to Monterey to meet someone else from Oregon, but I'm glad I learned about the CSIA program at SOU. Although I am not working with the SOU faculty, I am able to use the existence of the program at SOU to support the need for a similar program at our campus.

5. SUMMARY

To summarize the results of my participation in WECS:

- I am working to create a new security emphasis for our CS majors.

- I have taught the first course dedicated to security on our campus.
- I have significantly refined my presentation of security in my existing courses and had the opportunity to incorporate security into courses that I never considered before.
- I have discovered my own personal niche for teaching and learning: security.

I am very thankful for the opportunity to attend WECS. It has greatly expanded my interest in and awareness of security and security education. Because I am not a researcher with a large number of grants to draw on, the fact that all expenses were paid was key in making attendance a possibility for me. WECS was a first-rate idea that was executed flawlessly.

REPORT ON THE DEVELOPMENT OF CSN 290

The Creation of an Introductory Network Security Course

Marina A. Cappellino

Genesee Community College

Abstract: Approximately two years ago, I spoke with administration at Genesee Community College about attending Network Security training. The Director of Computing was considering expanding our technology programs to include at least one course- if not an entire degree- on the topic of Information Assurance (this includes Network and Information Security). Administration was supportive of this new initiative. At that point, I took the lead in developing an entirely new course entitled an *Overview of Computer and Network Security*.

The process of developing a course from the ground up was brand new to me. Some of the questions I had to addresses were: what would be considered important material? How could I boil down the masses of information received into a one semester introductory security course? How would I ascertain the level of proficiency of students? What would be the outcomes of the course? Could the material covered in this course be immediately implemented by students at their place of employment? Would abuse of college property in the forming of hacking be a concern? Would administration continue to support this effort through sustained funding for training and additional equipment needs? I hope to address these questions and more in the following report.

Key words: Network Security, Process of course development

1. INFORMATION ASSURANCE OBJECTIVES FOR THE 2005-2006 ACADEMIC YEAR

Upon the advice of the dean and looking at projected enrollments, we decided to offer the network security course in the Fall 2005 semester. This new topics course enrolled 10 students. Course objects, as required by a SUNY wide student learning outcomes initiative, were clearly stated in the syllabus as follows:

1. Explain at least two reasons why network security is important.

2. Demonstrate knowledge of basic network security principles.

3. Identify and describe at least two risk mitigation strategies

4. Describe the daily tasks involved with managing and troubleshooting computer security technologies.

5. Document knowledge of how to create a secure computer networking environment

6. Explain the concepts involving authentication, along with at least two of the types of attacks and malicious code that may be used against your network, and at least two countermeasures that could be taken against such attack.

7. Define terms and concepts such as intrusion detection systems, firewalls, and cryptography

8. Discuss at least two types of scanning and analysis tools

9. Discuss at least two legal and ethical issues within the field

2. WHAT WAS YOUR PROCESS IN INCORPORATING WECS6 MATERIAL AND HOW MUCH OF THE MATERIAL DID YOU INCLUDE IN THE COURSE?

I attended several training sessions, seminars and conferences on the topic of Network and Information Security, one of which was WECS6 training at the Naval Postgraduate School in Monterey. WECS helped me define the topics I would cover in my course and also gave me the opportunity to gain a better understanding of Network Security.

Upon learning that the WECS training included colleagues from four year colleges and universities, I was unsure of the amount of relevant information I would receive, which would be useful in an introductory community college course. I felt that perhaps the information would be targeted at a higher level. As I progressed through the WECS training and conference, my concerns were alleviated. There were a number of sessions that directly assisted me preparing for my network security course. Those sessions which presented material beyond the scope of my course, were invaluable as well. They afforded me the opportunity to gain a deeper understanding of network security. As a result, I went into my course with more confidence due to the newly gained knowledge.

While the book I selected, *Fundamentals of Network Security* by Eric Maiwald, covered an overview of many topics, there were certain topics I chose to cover in greater detail. For those topics, I would refer to many of my new resources for the clearest and most concise way of explaining the information. Professor Fulp's WECS notes were very helpful in the creation of supplemental slides on Firewalls. In addition, his session on Information Assurance Education Pedagogy was very interesting and extremely applicable- not only to my network security course, but to other technology courses I teach. I learned some helpful techniques for effectively teaching technology and the importance of a solid foundation in the basic concepts. His notes underscored the importance of clarifying for students, the terminology that can get so confusing so quickly. Professor Fulp spoke on the topics of "principle of least privilege", "hard on the outside vs. hard on the inside" and "defense-in-depth" all of which I covered in my course.

Professor Fulp's seminars were not the only ones from which I obtained a wealth of information and ideas. I also learned and incorporated concepts from Professor Dinolt's presentation on Cryptography. Seeing the code for the "I Love You" virus was very interesting and made my lecture on IDS signatures easier to understand. The labs designed for my course were a more basic version of some labs I worked with at WECS. I found the demonstration on

Steganography fascinating and incorporated this topic into my class as well- which the students very much enjoyed.

To reiterate, the benefits of the WECS training and conference were multifaceted. First, between 50%- 60% of my course is comprised of concepts I learned at WECS. I incorporated a sizable amount of material into my lectures, demonstrations and labs. I also obtained additional knowledge in Information Assurance from WECS, beyond which I taught in my own course.

Another invaluable part of my experience at WECS was the camaraderie I gained from colleagues of universities as well as colleges in Hawaii and Mexico. It is important to know the expectations of four year colleges with respect to knowledge of incoming students. I wanted to make sure that, in general, the material I hoped to cover in my course would be considered adequate introductory knowledge for a student eager to pursue Network Security at the university level. This topic was touched upon in my informal discussions with WECS participants from four year colleges and universities. The information gained and contacts obtained from four year institutions will be of even greater importance if Genesee Community College decides to embark on a degree granting program in Network Security/ Information Assurance.

3. ADDITIONAL EXPERIENCES AND SUMMARY

As mentioned above, what helped me in developing my course was the opportunity to travel to a variety of different seminars and training sessions. Though I ended up with a large amount of information at the end of each training seminar, there were certain topics that came up over and over again. These topics, I deemed as especially relevant to network security and made sure to cover them in my own course. Starting out, my quandary was "How do I figure out what is considered most important and how do I gear the information received to the intellectual level of my students?" Going to different seminars assisted me in developing answers to that question. Through the many seminars, workshops and conferences attended, the administration at Genesee Community College has been supportive every step of the way. In fact I doubt this endeavor would have been possible without the sustained support of the college.

I realize that assessing the different areas of security covered is an ever changing process that may take on a somewhat different form each time I teach the course. What was important and relevant in the area of network security even five years ago is completely different than what it is today. I anticipate that some topics I consider important today might not be so in years to come. Continued training is even more essential in Network Security than in other areas of technology.

One concern of teaching a network security course is the potential problem of hacking outside of a lab environment. This semester, our department instituted a policy in which **every** student enrolled in a Computer Systems and Networking course be required to sign a code of conduct form. We believed it was not wise to signal out one course (i.e. the network security course) for mandating the signing of a code of conduct form. Having **all** students in CSN sign this form, helped to emphasize responsible computing for every student, at every level and for every CSN course. We had no issues this semester with abuse. I believe this was partly due to the fact that students were well aware of the ramifications of such actions.

Having completed the course less than two weeks ago, I have no official written feed back from students. Students did however, express enjoyment and informed me that they had gained a great amount of knowledge from having taken the course. In the coming weeks, I will start to assess the course and lessons learned and begin to incorporate changes and enhancements. I anticipate the course will be offered again in the Fall 2006.

Index of Key Words

Notes